A Simple Model of Ocean Waves

Alain Fournier †

Department of Computer Science
University of Toronto
Toronto, Ontario

William T. Reeves

Animation Research and Development
PIXAR
San Rafael, CA

ABSTRACT

We present a simple model for the surface of the ocean, suitable for the modeling and rendering of most common waves where the disturbing force is from the wind and the restoring force from gravity.

It is based on the Gerstner, or Rankine, model where particles of water describe circular or elliptical stationary orbits. The model can easily produce realistic waves shapes which are varied according to the parameters of the orbits. The surface of the ocean floor affects the refraction and the breaking of waves on the shore. The model can also determine the position, direction, and speed of breakers.

The ocean surface is modeled as a parametric surface, permitting the use of traditional rendering methods, including ray-tracing and adaptive subdivision. Animation is easy, since time is built into the model. The foam generated by the breakers is modeled by particle systems whose direction, speed and life expectancy is given by the surface model.

To give designers control over the shape of the ocean, the model of the overall surface includes multiple trains of waves, each with its own set of parameters and optional stochastic elements. The overall "randomness" and "short-crestedness" of the ocean is achieved by a combination of small variations within a train and large variations between trains.

Rendered examples of oceans waves generated by the model are given and a 10 second animation is described.

CR Categories and Subject Descriptors: I.3.3 [**Computer Graphics**]: Picture/Image Generation - Display Algorithms; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling - Curve, surface, solid and object representations - Modeling Packages; I.3.7 [**Computer Graphics**]: Three-dimensional Graphics and Realism - Animation - Colour, shading, texture.

General Terms: Waves, Model, Algorithms.

Additional Keywords and Phrases: Ocean Waves, Wave Refraction, Surf and Spray.

†Address for 1985-1986: Department of Computer Science, Stanford University.

1. Introduction

There is no longer a need to justify the modeling of natural phenomena and no need to restate the seemingly limitless complexity of nature. One natural object especially interesting to model, and important because, to give only one reason, it covers two-thirds of the surface of the earth, is the ocean.

Compared to other natural objects and phenomena, the sea is both easier and more difficult to model. It is easier because it is relatively homogeneous, and more difficult because it moves. In fact, if it did not move, the surface would be flat, and we would be left with only a rendering problem (albeit not a simple one, due to water transparency and reflectance). Because of the homogeneity, we do not have to worry about widely different shapes (such as in species of trees) or different shaping mechanisms (such as in tectonic phenomena). We can hope to find a model that will be satisfactory for a wide range of sea conditions. Because the sea moves, the model has to include time explicitly in the surface description.

Looking at the sea (and this is where it should always begin and end), we can observe a wide range of "basic" familiar wave shapes, with typical crests and troughs, but it is hard to identify distinct individual waves or distinct trains of waves. In fact, individual waves are short-lived and all different, and the sea often appears to be made of randomly distributed waves. Our model must deal with this mixture of randomness and regularity.

We can, at the outset, establish a wish list of what we hope our model to accomplish. The model should:

- produce shapes similar to the shape of real waves;
- generate realistic motion;
- give a range of waves from very low ripples to high storm waves;
- make the surface take the shape of the shore;
- show the effect of shallow bottoms, such as refraction and breaking;
- identify the areas where breakers occur;
- allow an interface to a user-specified "wind field"; and
- include parameters to allow departure from "earth" conditions.

From the point of view of rendering, the model should:

- accommodate some, if not all, of the familiar rendering techniques;
- be easy to compute (in terms of basic operations and in terms of number of operations per point);
- permit rendering the surface at controllable levels of detail; and
- allow easy spatial and temporal filtering.

We also can list a few characteristics the model should not have:

- It will not be totally stochastic, even though stochastic elements are necessary.

In other words there is a strong deterministic element among the noise. Waves do roll in parallel to the shore and there are trains (the swell from storms thousands of miles away can be felt).

- It will not be a "fractal" model.

The reason given above is enough, but even considered as a random process the ocean surface does not have the characteristics of fractal noises. Figure 1 gives the energy spectrum of the sea. It is only a "generic" spectrum to give a qualitative idea. The figure also includes some of the typology of waves. It is obvious that it is not the spectrum of a fractal, which is characterized by a slow decay of the energy in the high frequencies. Nevertheless, fractals have been used to model water surface [Ogde85]. In fact, in the range we will attempt to model, that is waves

of periods from about 0.5 to 30 seconds, where the major disturbing force is due to the wind, and the major restoring force is due to gravity, the distribution is close to being gaussian. By the same token, the probability distribution for the surface displacement of the sea fits a gaussian very well.[1]

- It will not be a height field.

A height field, which has been used before to model waves [Max81, Scha80], is such that for each pair (x,y) there is at most one z value. Figure 2, which is realistic in its own way, shows us why height fields should be avoided.

- There should not be a net transport of matter as time advances.

We do not want the whole sea, as an object, to march across the landscape. While we certainly want local motion, as in the water lapping against the shore, the sea should globally stay put, as it fortunately does most of the time. There can be a transitory net mass transport, but we

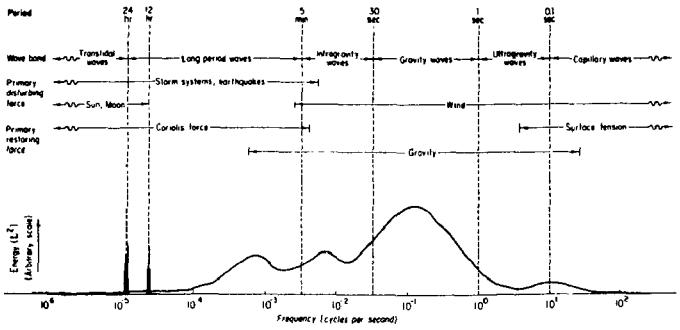

Figure 1
Rough energy spectrum of the sea (from [Kins84]).

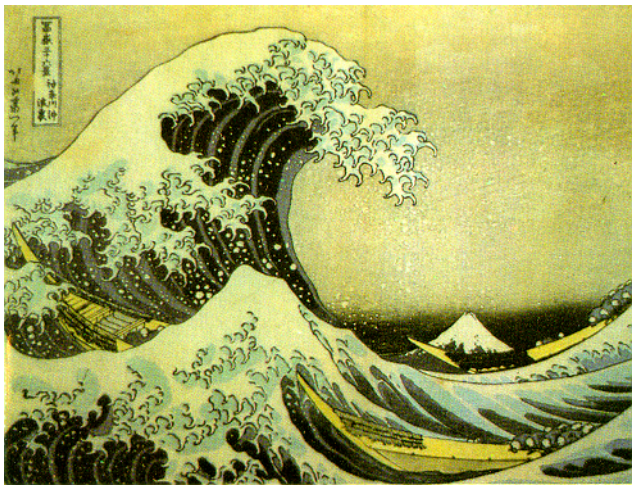

Figure 2
The surface of the sea is not a height field.

will neglect this effect.

- We do not expect any "physical" answer from our model.

In fact, we do not ask any question except "does it look like the real thing". So a model which might be useless to an oceanographer might turn out to be good for our purposes. By the same token, we will feel free to adjust coefficients to achieve the needed effect, even if we are not quite sure if doing so has a physical meaning. This is of course extremely bad form when building physical models.

1. The distribution of the heights of wave crests fits a Raleigh distribution (*cf* [LeMe76], Appendix A). The distribution of wavelengths is not so easily expressed.

2. Sources and Previous Work

In the time honored tradition of computer graphics, we can now go ahead and steal anything we can get. The literature of hydrodynamics and waves is quite large, but we relied mainly on the books of Crapper [Crap84], Le Mehaute [LeMe76], Leblond [Lebl78] and Kinsman [Kins84], which is a Dover reprint of [Kins65]. The latter book by B. Kinsman, *Wind Waves*, is highly recommended. It is a rarity, a book filled with equations but written by somebody who obviously loves words and uses them superbly.

There have been a few previous attempts to model waves or some aspects of the ocean surface before in computer graphics. The first two are referred to by their appearance in the Siggraph slide sets, since they were not described elsewhere to the best of our knowledge. The first is the "Pyramid" slide, by Gary Demos *et al*, in the 1981 collection. It appears to use cycloidal waves and it was also animated (see reference in [Max81a]). The other is the "Night Castles" slide, by Ned Green, in the 1982 collection which used sine waves and the "bump mapping" technique [Blin78]. Nelson Max, in his famous *Carla's Island* [Max81b], also used a few terms of a sum of sine waves to approximate Stokes waves [Max81a]. His chosen rendering algorithms required a height field. That was accompanied by partial ray-tracing and imaginative sunset and moonlight effects. Norton, Rockwood and Skolmoski [NoRS82] used frequency limited ("clamped") analytical functions to model waves. The technique was intended for use in a real-time image generator. Schachter [Scha80] used the sum of narrow band noise waveforms, a technique also intended for real-time applications, and which was implemented in hardware.

More recently Perlin [Perl85] describes various types of waves as part of an application of his Pixel Stream Editor. He used, among others, spherical cycloidal waves and, specifically to simulate sea waves, random sources with a constant amplitude-frequency product. Ts'o and Barsky [TsBa85] model the sea surface as a height field from the Stokes model, and fit it with β-splines, adjusting the shape parameters to obtain symmetrical waves. They were the first to have as one their goals the modeling of wave refraction as the ocean floor depth decreases. Peachy [Peac86] models the ocean surface using a height field computed from the Stokes model, with quadric surfaces to introduce asymmetry. He also used particle systems to model the surf, but did not include depth effects.

3. The Basic Model[2]

Fluid motion is generally studied by one of two methods. In the *Eulerian* method, one considers a point (x,y,z) and tries to answer questions about the properties of the fluid at this point as a function of time, such as the speed:

$$U=f(x,y,z,t)$$

In the *Lagrangian* method, one follows the trajectory of a point (x_0,y_0,z_0), given by a reference position. This can be seen as the trajectory of a particle of matter. For instance, we can ask for the speed at time t:

$$V_x=f_x(x_0,y_0,z_0,t)$$
$$V_y=f_y(x_0,y_0,z_0,t)$$
$$V_z=f_z(x_0,y_0,z_0,t)$$

The first method is favored in hydrodynamics, and in the study of waves, especially since the development of stochastic models for the analysis of the sea. The second method looks *a priori* more useful for graphics modeling. It is more convenient to treat the ocean surface as a graphical primitive and a Lagrangian approach will describe it explicitly.

We will consider that each particle (i.e., each point) on the free surface describes a circle around its rest position (x_0,y_0,z_0). Without loss of generality, we will orient our world coordinates so that the XY plane is the plane of the sea at rest, and the Z axis is pointing up. For the time being, we will only consider the motion in the XZ plane. The equation of the motion of a particle is then:

$$x=x_0+r\times\sin(\kappa x_0-\omega t) \quad (1a)$$

$$z=z_0-r\times\cos(\kappa x_0-\omega t) \quad (1b)$$

Looking at the equations above as parametric equations in x_0 for a given t and a constant z_0, one can see that the surface is a *trochoid*, a generalization of a *cycloid*. It can be seen as the curve generated by a point P at a distance r from the center of a circle of radius $\frac{1}{\kappa}$ rolling over a line

2. A glossary at the end of this paper lists the symbols used.

at distance $\frac{1}{\kappa}$ under the X axis (Figure 3). The parametric equations are, for t=0 and z_0=0:

$$x=-\frac{\alpha}{\kappa}-r\times\sin(\alpha)$$

$$z=-r\times\cos(\alpha) \quad \text{where} \quad \alpha=-\kappa x_0$$

The height of the wave is H=2r, the wavelength is $L=\frac{2\pi}{\kappa}$, the period is $T=\frac{2\pi}{\omega}$ and the phase speed (the speed of travel of the crest, for instance) is $c=\frac{L}{T}=\frac{\omega}{\kappa}$. The phase is $\phi=\kappa x_0-\omega t$, assuming a phase of 0 for x_0=0. Figure 4 shows the various shapes obtained for κr=0.2 which is almost a sine wave, for κr=0.7, κr=1.0 which is a cycloid, and κr=1.2, which leads to self intersection. In fact, the limiting reasonable value for κr is 1, which means that $\frac{H}{L}=\frac{\kappa r}{\pi}=\frac{1}{\pi}$. The ratio $\delta=\frac{H}{L}$, called the *steepness* of the wave, functions as the *shape parameter* of the wave.

The closed circular orbit is not merely a convenient description. It can be derived from the equations of motion for deep water small amplitudes waves [Kins84, LeMe76, FeLS65]. In fact this model was proposed long ago in oceanography, and is called the Gerstner's or Rankine wave [Gers1802, Rank1863], see also [Kins84]. It is seldom used in contemporary oceanography, because it is hard to extract information from it, but it is satisfactory for our purposes. Note that there is no net mass transport.

So this is our basic model. To introduce the needed effects and variations, we will manipulate the parameters of the orbit equations (modifying the radius, the phase angle, turning the circle into an ellipse, etc.).

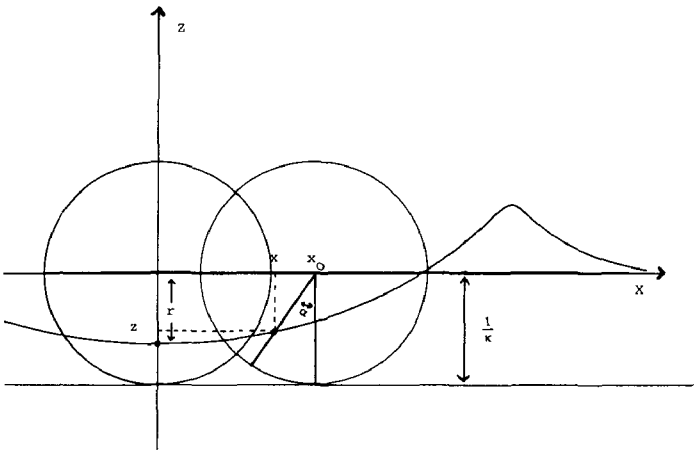

Figure 3
The trochoid.

Most of these parameters are physically related. In deep water (we will see later that *deep* is relative to the wavelength), there is a simple relationship between the period and the wavelength:

$$L=\frac{gT^2}{2\pi} \qquad (2)$$

which can be stated as:

$$c=\frac{gT}{2\pi}$$

The group velocity (the speed at which the energy carried by the wave travels) is then $U=\frac{c}{2}$. These relations are valid only when the basic differential equation for the flow of water can be simplified to be linear, but this is a good approximation for slow motion. We can go a little further, and relate the height of the wave to the wind speed, to provide reasonable defaults. Here again we have to simplify tremendously. If a wind of speed V has been blowing for ever over an infinite *fetch* (the area covered by the wind), we obtain a *fully aroused sea*. One possible formula for the *significant height* of the waves (the average height of the highest one-third of the waves) is ([Kins84]):

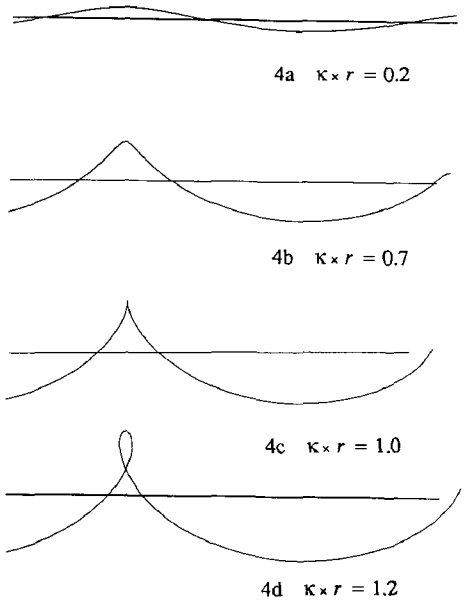

Figure 4
Shapes of waves as κr varies.

$$\overline{H}_{\frac{1}{3}}=7.065\times10^{-3}V^{2.5}$$

The heights are in meters, and the speeds in meters/seconds. We can take this height to be representative, and create a wave of that height if only the wind speed is given. In this case, the average L and T are related by the formula in equation (2), and the frequency at which the maximum in the energy spectrum occurs is given by:

$$\omega=\frac{g}{V}\sqrt{\frac{2}{3}}$$

Thus, only a wind speed has to be supplied to completely define the waves under these simplified conditions. It is important to note that these different relations do not strictly apply to the same definitions of *average*. We can use them only because we are looking for "reasonable" values, and we will not base numerical forecast on them.

4. Special Effects

The first modification to the basic equations (1a) and (1b) is a trivial one. It is to take the direction of the wave into account. Initially the wave fronts are in a direction perpendicular to the wind that created them, and the plane of the orbits is defined by the wind speed vector and the Z axis. Wave fronts keep the same direction even after the wind ceases or blows in another direction. To impose a direction, we perform a two-dimensional rotation on the orbit plane. In this section, we will stay in the original coordinate system, so that the waves progress in the positive X direction, and the crests are parallel to the Y axis.

The second modification can properly be called a *kludge*. To model the effect of the wind on the top of the crests, a third term is added to the expression for the phase angle ϕ, proportional to the height of the wave above the sea level at rest (with the sign included). Thus the phase angle becomes:

$$\phi=\kappa x_0-\omega t-\lambda\Delta z\Delta t$$

The term Δz is $z-z_0$. The term Δt is included because this effect is proportional to time. λ is the coefficient of proportionality. This is a temporary effect on the phase, which does not accumulate, and sums to 0 both in time and in space. It maintains the shape of the orbit but causes the point to accelerate at the top and decelerate at the bottom. The shapes of the waves obtained with various λ are given in Figure 5.

The next modification is to account for the effect of the depth h at the point (x_0, y_0, z_0) measured positively down from the sea level at rest. The first effect of the depth is to alter the wavelength of the wave. It is assumed, and largely true, that the period is not affected. If we call κ_∞ the wave number at infinite depth, a good approximation for the wave number κ at depth h is:

$$\kappa\tanh(\kappa h)=\kappa_\infty \qquad (3)$$

It is unfortunately a transcendental equation in κ. But when $x\to 0$ then $\tanh(x)\to x$. Therefore at small depth (which means when κh is small), the relation becomes:

$$\kappa^2 h=\kappa_\infty \text{ or } \kappa=\sqrt{\frac{\kappa_\infty}{h}}$$

When $x\to\infty$, then $\tanh(x)\to 1$, so $\kappa\to\kappa_\infty$, which is to be hoped. Note that since $\kappa h=\frac{2\pi h}{L}$, a ratio $\frac{h}{L}$ of 0.5 gives an argument of π for the tanh which makes it practically equal to 1. Thus for practical purposes, "deep" is related to the wavelength, and means a ratio $\frac{h}{L}$ greater than 0.5. A good approximation for equation (3), valid for the entire range within 5% is:

$$\kappa=\frac{\kappa_\infty}{\sqrt{\tanh(\kappa_\infty h)}}$$

Since the wavelength is affected, so is the phase speed, which means: $\frac{c}{c_\infty}=\frac{\kappa_\infty}{\kappa}$ and the waves are refracted as they slow down. In fact one can apply Snell's law:

$$\frac{\sin(\alpha_h)}{\sin(\alpha_\infty)}=\frac{c_h}{c_\infty}$$

to compute the angles the wave fronts make when going from infinite depth to a depth of h (or between any two depths for that matter). Figures 6, 7 and 8 illustrate the refraction of waves caused by two bottoms of constant depth, a gently sloping beach, and an undersea valley. A *decay factor* multiplies the argument of the tanh function; it scales the real depth to control the intensity of the effect. Another parameter allows the user to clamp the effect and limit the shortening of the wavelength.

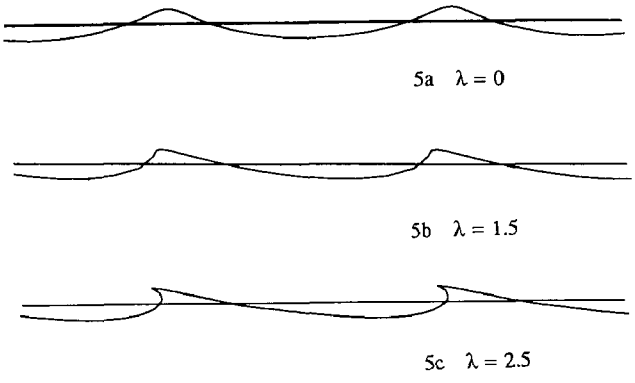

Figure 5
Shapes of waves for various λ.

The depth effect cannot be computed from local information only, since the phase delay it introduces is cumulative. Now κ is a function of the depth, which in turn is a function of x_0 (and of y_0 in the general case). Assuming that $\phi=0$ for $x_0=0$, and for brevity that λ is zero, the phase equation should now be:

$$\phi=-\omega t+\int_0^{x_0}\kappa(x)dx \quad \text{where} \quad \kappa(x)=\frac{\kappa_\infty}{\sqrt{\tanh(\kappa_\infty h(x))}}$$

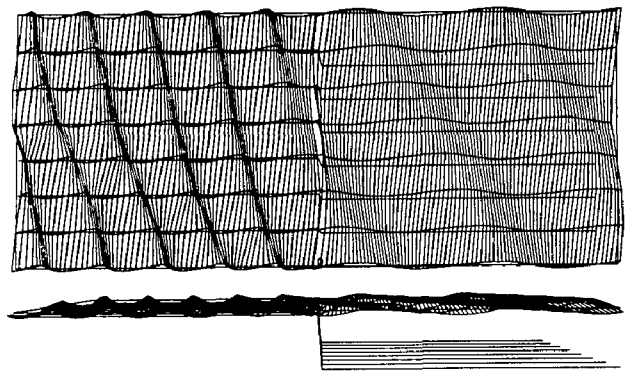

Figure 6
The refraction of waves.
The angle with the X axis is 30° on the deep bottom.
On the shallow bottom to the left the wavelength has been halved.

Since in general we do not even know h(x), an exact solution is not very likely. Furthermore, when considered in two dimensions, the integral along the path followed by the wave varies according to its direction. So precomputing this integral is not practical. Since we want to be able to have several trains of waves passing over a given point, remembering the cumulative depth information at each point for each train would be costly in storage. Even more seriously, we want to be able to compute points in any order and at any spacing, so we do not want to depend on the phase value passed by the neighbors.

The solution is to have the *trains* themselves accumulate and carry the information as they "sweep" the bottom. Each train carries a grid (of a resolution chosen to pick up the rough features of the bottom), and for each point of that grid at position x the term:

$$\kappa_\infty\left(-1+\frac{1}{\sqrt{\tanh(\kappa_\infty h(x))}}\right)\Delta x$$

is added. So the cumulative term D is:

$$D=\sum_0^{x_0}\kappa_\infty\left(-1+\frac{1}{\sqrt{\tanh(\kappa_\infty h(x))}}\right)\Delta x$$

When computing ϕ for a given x_0 for this train, it is evaluated as:

$$\phi=\kappa_\infty x_0-\omega t+D$$

which is equal to:

$$\phi=-\omega t+\sum_0^{x_0}\frac{\kappa_\infty}{\sqrt{\tanh(\kappa_\infty h(x))}}\Delta x$$

Thus we have a good approximation of the integral. The cumulative term

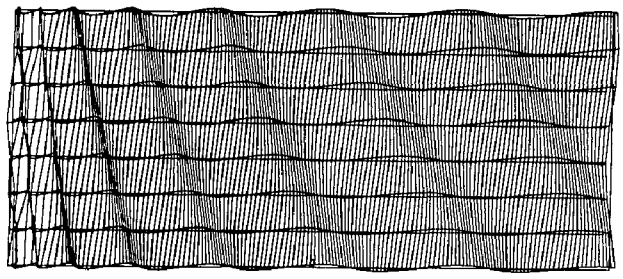

Figure 7
Refraction of waves.
The beach gently slopes down from the left.
The wave fronts align themselves with the beach.

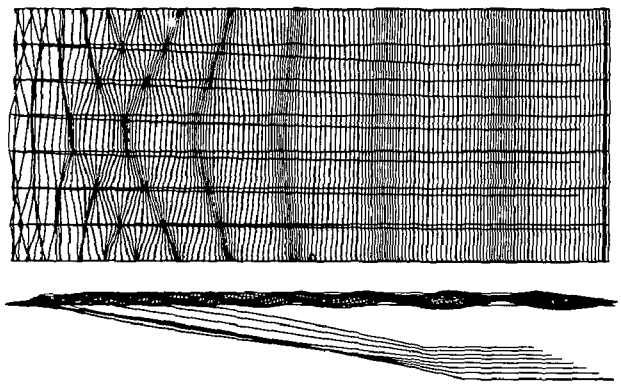

Figure 8
Refraction of waves over an undersea valley.

for x_0 on the sea surface is bilinearly interpolated from the nearest cumulative terms of the train grids. Trains will be described in more detail in the next section.

The last modification models the waves breaking on the shore. Classic theories predict that the orbits will become ellipses in water of intermediate depth, and eventually collapse down to straight line segments as $h \to 0$ (see Le Mehaute, p. 235, for example). More interestingly, Biesel [Bies52], proposed a model where the major axis of the ellipses tends to align itself with the beach as the depth goes to zero. This effect gives a very realistic shape to the breakers. Unfortunately his coefficients are fairly complicated and costly to compute (even though they are only functions of κh, and could be determined by lookup tables). We decide to adopt our own model, which keeps most of the necessary features but is simpler and easier to control. Equations (1a) and (1b) are replaced by:

$$x = x_0 + r \times \cos\alpha \times S_x \times \sin\phi + \sin\alpha \times S_z \times \cos\phi$$

$$z = z_0 - r \times \cos\alpha \times S_z \times \cos\phi + \sin\alpha \times S_x \times \sin\phi$$

ϕ is defined as before, $\sin\alpha$ is given by $\sin\alpha = \sin\gamma e^{-\kappa_o h}$, where γ is the slope of the bottom in the direction of the wave travel. S_x and S_z are:

$$S_x = \frac{1}{1 - e^{-\kappa_x h}} \quad \text{and} \quad S_z = S_x(1 - e^{-\kappa_z h})$$

The κ_o, κ_x and κ_z terms are scaling constants.

The net effect is to flatten the circles into ellipses (down to line segments when $h=0$), to orient their major axis toward the slope of the bottom, and to increase the length of the major axis. Figure 9 shows the influence of this effect on the orbitals, and Figure 10 shows an example of the wave shapes obtained.

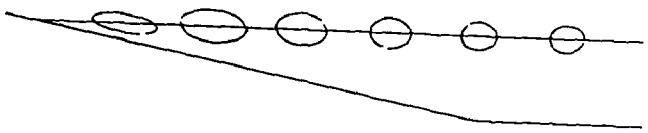

Figure 9
How the depth affects the orbits.
Gaps represent points at the same time.

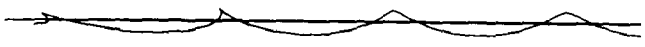

Figure 10
How the depth affects the shape of the waves.

It should be noted that some models predict a slight decrease in radius before it starts increasing. We decided to neglect this. It is also important to note that $S_x \to \infty$ when $h \to 0$. As before, parameters allow the users to control this effect and clamp them.[3]

5. Breakers, Spray and Foam

One can distinguish two types of breakers, a *plunging* type, which appears on waves whose shapes are similar to the one in Figures 2, 5 or 10, and a *breaking* type, for waves like in Figure 4b and 4c. The exact conditions for breakers are hard to specify, but the consensus seems to be that to cause breakers:

- The particle speed has to be greater then the phase speed, and
- the curvature of the surface has to be high.

Some other conditions are occasionally mentioned, notably that the slope has to be "nearly vertical", as a sufficient condition. A paper by Longuet-Higgins [Long76] contains a picture which proves that it is not so.

It is interesting to note that our model is "aware" of the particle speed, which gives it a better chance to test for breakers, and make informed decisions about what to do. We have experimented with various conditions, but the simplest and the most reliable (as checked by our intuition of where breakers should appear) was to test the curvature, and decide a breaker should appear if it is above a set threshold. In practice a curvature roughly corresponding to an angle of 60° at a surface point seems reasonable.

When a breaker is detected, spray and foam are generated. Spray is generated if the difference between the particle speed and the surface speed projected in the direction of the normal to the surface exceeds a set threshold. The spray is then sent in that direction and follows a trajectory affected by gravity after that (taking air drag into account). It is rendered by particle systems (see section 7).

If the speed threshold is not passed, then foam is sent sliding along the wave surface. Each "unit" of foam is given a lifetime, and is passed from point to point, taking the acceleration of gravity and viscosity into account, until it "dies." The foam is also rendered by particle systems. Figure 11 illustrates the mechanism for foam. Breakers occur on the crests in the lower left hand side. Plus signs indicates points where breakers are generated in this particular frame, and "o"s indicate where there is foam which has been generated earlier. In the case of the second crest, the sources of foam move towards the bottom right. For the first crest, they remain relatively fixed with respect to the crest. The "S" characters indicate where the particle speeds are high enough to generate spray.

Figure 11
Detection of breakers and spray, and generation of foam.
At "+" breakers are detected, at "S" spray
is generated, at "o" foam is "alive".

6. Wave Trains and their Properties

We have described a model for regular, long-crested waves. We have now to introduce the variability and randomness characteristic of the real sea.

Wave trains define groups of similar waves that permit controlled variation both within trains and between trains. A wave train[4] is a rectangular box on the surface of the sea containing waves of the same basic characteristics and of the same phase origin. Thus the heights, periods and

3. The model requires the computation of many transcendental functions. Fortunately, they are all in the range 0-1, and all can be computed through lookup tables.

4. A wave train is not to be confused with a *wave packet*. The latter is a convenient mathematical object to study sums of sine waves.

wavelengths of waves within a train are originally the same. The local coordinate system is always such that the direction of motion is towards the positive X axis, the crests are parallel to the Y axis, and the phase is 0 at the origin. The train as a whole has an origin in the world coordinate system, a direction (given by an angle positive counter-clockwise from the world X axis), a size in X and Y, and a speed (as distinct from the speed of the waves it contains). By default the train speed is half of the phase speed, that is, equal to the *group velocity* in deep water. The trains can also be turned off and on if necessary.

To shape the train (rectangular boxes are rare on the sea) an amplitude function gives each train its own *height envelope*. The function takes the local coordinates and returns a local radius (i.e., height). For most trains, the amplitude function is the product of an X-amplitude function and a Y-amplitude function. In the Y direction the function is usually an envelope with a flat top and soft decay on the side. In the X direction, the same function is multiplied by a random component to vary the wave heights within the same train. Figure 12 gives a typical example of a X-amplitude envelope. Of course, other distributions can be used (see footnote 1).

The variation in wavelengths within a train is obtained by transforming from the local x_l to an adjusted value, x_a, such that $\frac{dx_a}{dx_l}>0$ and the boundary values are the same. Those two effects, variations in height and variations in wavelengths can be made a function of time, so that the "age" of the train is taken into account, although we have not implemented this effect.

To further reduce the regularity of the wave trains, random "holes" are distributed within the trains to interrupt the crests. These holes, whose number is determined for each train by the user, have a diameter on the order of the wavelength, have an inverted bell-shaped height profile, and have a *Poisson disc* distribution.

The trains also carry the cumulative depth effect described in section 4. A grid is defined for each train, and at each time update of the train the incremental depth effect is calculated, interpolating the depth at this point from the terrain grid points (or depth function if there is one). A time decay factor is optional in the cumulative depth factor, to allow for a slow decay of the bottom effect as the train moves on. The train grid can be rather coarse, as the depth effect does not have to give a sharp image of the bottom profile. However, too coarse a grid would cause rather brutal slope discontinuities. Also, interpolation from the grid might make the waves feel the depth ahead of their current position. This can be avoided by not using the train grid points ahead of the waves for interpolation, at a slight cost in precision on the interpolated cumulative term. To initialize the train grid for the depth effect, trains can be backed up a given distance before starting the wave computations, so that they have accumulated enough depth information.

In summary, we compute the position of a point on the ocean surface as the sum of the displacements from the point's rest position due to all active trains covering this point. Our assumption of a linear addition law is accurate enough for our purposes, especially since we do not add the surface displacements, but the Lagrangian displacements.

To create a model, the designer specifies the number of trains and model characteristics for each train. An interactive two-dimensional version of the model demonstrates the effects of the selected parameters. A train with small waves and a high degree of randomness is usually selected for a noisy background, then two or three trains with sizeable wave heights generate the interesting features. It is interesting to note that "normal" waves have a fairly low steepness (δ from 0.008 to 0.1), and only about 10% of the waves are above 6 meters in height [Kins84]. One usually has to exaggerate heights and steepness to make things more spectacular.

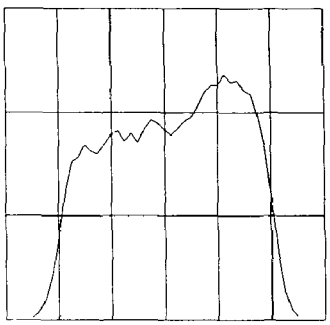

Figure 12
A typical X-amplitude envelope.

The use of trains as controllable groups of waves has another advantage. We can easily model the reflection of waves on walls, rocks or high slope shores by generating a second train for the "mirror" image of the reflected one, with the height of the waves reduced as a function of the slope of the shore.

7. Rendering Issues

Our model of ocean waves describes the three-dimensional geometry of the water surface and how it transforms and moves over time. We now wish to render that data into a sequence of color-shaded raster images that resemble as closely as possible the appearance of the real ocean. The surface generated by the model is a parametric surface whose parameters are the (x_0,y_0) of the sea surface at rest, and time. Points on this surface can be computed independently of their neighbors, so a regular grid is not needed, and ray intersection operations, adaptive sampling, and other point driven methods can be applied directly to the surface. Each point on the surface also has a natural bounding volume in the sphere centered at that point and whose radius is the sum of the radii defined by all the waves trains. We had available, however, an outstanding set of rendering software, and it made the rendering considerably easier. We used an anti-aliasing, scan-line oriented, adaptive subdivision rendering system called *reyes* [Cook86] that incorporates a sophisticated shading and texturing system using shade trees [Cook84].

Our first step in rendering a frame is to execute the model at a rectangular grid of points in the x–y plane in the world coordinate system. We then transform the resulting three-dimensional data points describing the surface of the water into a set of three-dimensional surface primitives. Currently, we can interpolate the data with either bi-linear or bi-cubic (Catmull-Rom) patches. Bi-linears are better when debugging and planning a scene as they are faster to compute and use less storage space. Bi-cubics are better for final images as the surfaces are smooth, not faceted. Selecting the right size of grid (and hence the number of patches) is important; we want enough resolution so that the waves in the train with the shortest wavelength do not alias, and yet we do not want so many patches that the scene will take forever to render.

The color and shading of the real ocean surface is derived mainly from the sky. In general, the ocean reflects its surrounding environment. To simulate this, we use an *environment map*. An image of the surrounding world is placed into a series of texture maps that together form a gigantic sphere about the waves. To determine the color of a point on the wave's surface, we perform what amounts to a one-level ray trace, bouncing a ray from the camera position off the surface point according to the rule: the angle of incidence about the surface normal is equal to the angle of reflectance about the surface normal. Where the reflected ray now intersects the enclosing sphere, which is very easy to compute, gives us indices into one of the texture maps and hence a color to be reflected.

The environment map used in Figure 13 models a very simple sky, and is made from a series of ramps going from dark blue (opposite the sun) to orange (behind the sun) and a very bright yellow disk to represent the sun. Clouds were painted unto the sky around the sun. More elaborate environment maps including shorelines and general terrain are possible and easy to add.

The one flaw in this scheme is that the ray trace is only one level. It only intersects the environment map and not other parts of the scene or other parts of the water itself. Nevertheless, it makes interesting pictures and is definitely less expensive computationally than general ray tracing.

The actual shade used for a given point on the surface is:

```
finalcolor = REF*reflcolor*refl(normal, viewer) +
    watercolor*(AMB*ambient() + DIFF*diffuse(normal) +
        SP*specular(normal, viewer, ROUGH));
```

where *normal* is the surface normal, and *viewer* the vector from the camera to the surface point. The *refl* routine returns the color from the environment map as reflected off the surface. The shading routines *ambient*, *diffuse* and *specular* return the intensity and color of the ambient, diffuse, and specular components from any light sources in the scene. *Reflcolor* and *watercolor* are two scaling parameters (they are RGB triples) that can be used to selectively filter out different parts of the color spectrum. For example, setting *reflcolor* to (1.,.9,.8) filters out some of the green and blue components of the reflected color. The constants *REF*, *AMB*, *DIFF*, and *SP* are other scaling parameters and *ROUGH* sets the roughness or sensitivity of the specular function. Typi-

cal values are: *REF=.8*, *AMB=.01*, *DIFF=.1*, *SP=.5*, and *ROUGH=.02*. For the *diffuse* and *specular* functions, we position a light source at the sun. The Fresnel coefficient for reflectance of water is included in *refl()* and *specular()*.

To model water shimmering and other forms of small perturbations like wave chop, we could use several wave trains with very small wavelengths and highly stochastic amplitudes. But this approach requires a very small grid size to represent all the surface detail and greatly increases the number of patches to render. Instead, we use *bump mapping* [Blin78]. The normal to the surface is slightly perturbed at each point when shading it. This affects the *diffuse*, *specular*, and *refl* components of the above shading equation and makes the surface appear more complex than it really is. The amount of perturbation is specified by a *bump map* texture which is mapped onto the surface. For animation purposes, the bump map must move with the wave surface in a natural way. To do so, we actually translate the *bump map* over time just like we translate trains over time.

Figure 13 was computed at 2048 by 1228 pixel resolution. It was modeled with 92,724 bi-cubic patches and took 10 hours and 21 minutes on a Computer Consoles Power 6/32 computer. Figure 14 used 85680 patches and took 2 hours and 21 minutes. To give an idea of modeling time, a 2,400 point grid, which when rendered is modeled with the same number of bicubic patches, takes about 36 seconds per frame to generate on a VAX 11/750.

8. Conclusions

We have shown that a simple basic model can produce a large range of wave shapes and phenomena. This includes the effects of depth such as refraction and surf, and some of the effects of wind. The model also has the advantage of fitting in well with most rendering methods, since it is essentially a parametric surface in two spatial parameters and one time parameter. The model also makes possible the detection of the appearance of spray and foam. The model has been easily interfaced to a completely independent rendering package, to produce the color pictures

Figure 13
Beach at sunset. Notice the breaking of waves on the shore, and how the crests take the shape of the shore.

All of the above shading calculations are handled with a shade tree in the *reyes* system. It also handles hidden surface and motion blur automatically.

The position, size and speed of each foam and spray "element" is passed to a particle system based program. For the spray, the particles follow the free fall laws of motion. For the foam, the motion is controlled by the model itself, as described above.

One final note about how we modeled the terrain in our images. With a tablet and a simple program, we entered a set of unequally spaced, three-dimensional data points. A surface (a mesh of patches) was then passed through these points using a localized version of Shepard's method [Shep64]. This same data set was used to specify the depth of the ocean floor.

Figure 13 is actually a frame from a ten second animation we are in the process of computing. The most satisfying feature of the sequence is the convincing way the breakers crash and the waves lap up on the beach. Figure 14 is an example of wave refraction due to the ocean floor shallowing near shore. Note how the waves tend to align themselves with the shoreline. The slope of the beach is more gentle on the right hand side, causing more shortening of the wavelength.

Figures 15 and 16 are examples of spray and foam generated by particle systems. Figure 16 shows what happens when the wind factor λ is used to obtain overhanging waves.

illustrating this paper. Few parameters and the inclusion of individually controlled trains of waves allow the designer and the animator to achieve the wanted effects relatively easily. We therefore met most of the goals listed in section 1, with the possible exception of the one concerning simplicity and a small number of the operations. We already mentioned that the square roots, trigonometric and hyperbolic functions can easily be replaced by lookup tables or *ad hoc* approximations. The number of operations per point, though quite large, is not prohibitive, as shown by the time necessary to compute frames.

Some of the effects which where not included, but are relatively easy to add within the model, are the creation of *wakes*, the *ageing* of wave trains, that is the modification in heights and wavelengths they undergo after their creation, and some diffraction. Other effects, such as modeling turbulent flow, additional phenomena due to refraction such as caustics, and the effect of moving ocean floors, would be mostly beyond our model, but then again some of these phenomena do not have very satisfactory physical models either.

We will have reached our goal if, by designing and implementing this basic tool to model many aspects of the ocean, we have included enough features to make it useful, and we have left out enough phenomena so that we will all be challenged to do better.

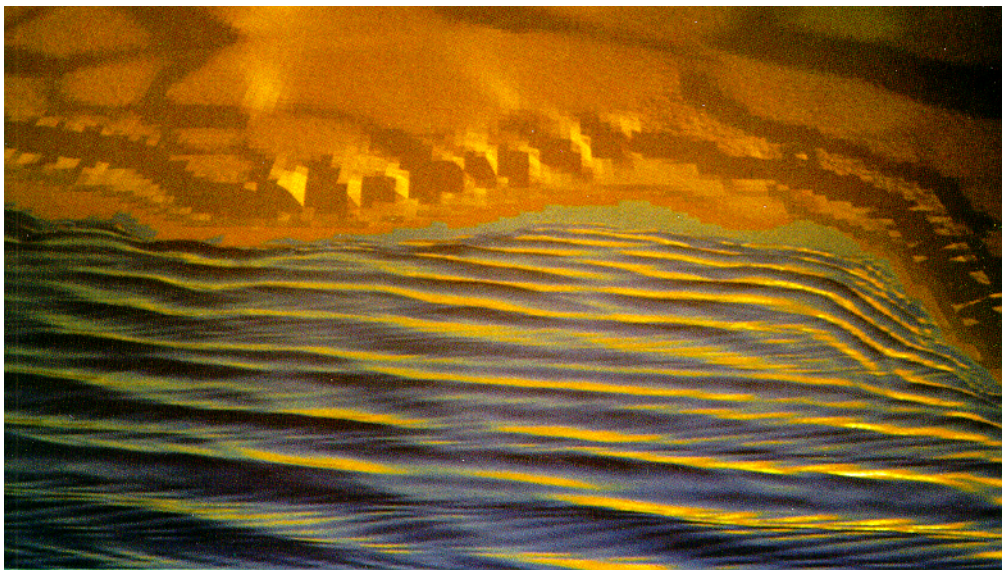

Figure 14
Wave refraction.

Figure 15
Waves with spray and foam

9. **Acknowledgements**

Most of this research was performed while Alain Fournier visited and Bill Reeves was employed by the Computer Division of Lucasfilm Limited.

Alain Fournier wants to acknowledge the financial support of Canada's National Science and Engineering Research Council, the hospitality of the Department of Computer Science at Stanford University, and the hospitality of PIXAR during many phases of this project. He also wishes to thank Adrienne, who took him to the ocean.

Bill Reeves wishes to acknowledge Rob Cook for his *reyes* rendering software, Eben Ostby for help with the environment maps, and John Lasseter for the clouds and advice on coloration.

Figure 16
Beneath the Waves of San Rafael

10. Glossary

Symbol	Meaning	Unit	Remarks
-			
L	wavelength	meter	
H	height of wave	meter	
δ	steepness	-	$\delta=\frac{H}{L}$
r	radius of orbit	meter	$H=2\times r$
T	period	second	
κ	wave number	radian/meter	$L=\frac{2\times\pi}{\kappa}$
ω	angular speed	radian/second	$T=\frac{2\times\pi}{\omega}$
c	phase speed	meter/second	$c=\frac{L}{T}$
h	depth	meter	
x_0,y_0,z_0	rest position	meter	
x,y,z	position at time t	meter	
g	acceler. of gravity	meter/second^2	vector
U	group velocity	meter/second	vector
V	wind speed	meter/second	vector
t	time	second	
γ	slope of bottom	-	in direction of wave
ϕ	phase angle	radian	
λ	wind factor	radian/m. $\times$ second	
D	cumulative depth effect	radian	

11. References

Bies52 Biesel, F., "Study of Wave Propagation in Water of Gradually Varying Depth," *in Gravity Waves*, U.S. National Bureau of Standards Circular 521, (1952), pp. 243-253.

Blin78 Blinn, J. F., "Simulation of Wrinkled Surfaces," in *Proceedings of SIGGRAPH '78*, also published as Comput. Graphics, *12*, 3, (Aug 1978), pp. 286-292.

Cook84 Cook, R. L., "Shade trees," in *Proceedings of SIGGRAPH '84*, also published as Comput. Graphics, *18*, 3, (July 1984), pp. 223-231.

Cook86 Cook, R. L., "Antialiasing by Stochastic Sampling," accepted to appear in *Transactions on Graphics*, (summer 1986).

Crap84 Crapper, G. D., *Introduction to Water Waves* (Chichester, West Sussex, England : 1984).

FeLS65 Feynman, R. P., Leighton, R. B. and Sands, M., *The Feynman Lecture Notes on Physics*, (Addison-Wesley, 1965).

Gers1802 Gerstner, F. J. v, "Theorie der Wellen," Abh. d. k. bohm. Ges. d. Wiss. *Also reprinted in* Ann. der Physik, 32, (1809), pp. 412-440.

Kins65 Kinsman, Blair, *Wind Waves* (Prentice-Hall, 1965).

Kins84 Kinsman, Blair, *Wind Waves* (Dover, 1984), *reprint of preceding.*

LeBl78 LeBlond, Paul H., *Waves in the Ocean* (Amsterdam, 1978.)

LeMe76 Le Mehaute, Bernard, *An Introduction to Hydrodynamics and Water Waves* (New York, Springer-Verlag, 1976)

Long76 Longuet-Higgins, M. S., "On Breaking Waves," *in Waves on Water of Variable Depth*, Provis, D. G. and Radok, R., *Eds*, Springer-Verlag Lecture Notes in Physics, (1976), pp. 129-130.

Max81a Max, N., "Vectorized Procedural Models for Natural Terrains: Waves and Islands in the Sunset," in *Proceedings of SIGGRAPH 81*, also published as Comput. Graphics, *15*, 3, (Aug 81), pp. 317-324.

Max81b Max, N., "Carla's Island," appeared in Issue #5 of the *SIGGRAPH Video Review*, (1981).

Mei83 Mei, Chiang C., *The Applied Dynamics of Ocean Surface Waves* (New York, c1983.)

NoRS82 Norton, A., Rockwood, A. P. and Skolmoski, P. T., "Clamping: A Method of Antialiasing Textured Surfaces by Bandwidth Limiting in Object Space," in *Proceedings of SIGGRAPH 82*, also published as Comput. Graphics, *16*, 3, (July 82), pp. 1-8.

Ogde85 Ogden, J., "Generation of Fractals Using the Burt Pyramid," *presented at the* 1985 Optical Society of America Annual Meeting, Washington, DC, (October 1985).

Peac86 Peachy, D., "Modeling Waves and Surf", *these Proceedings*.

Perl85 Perlin, K., "An Image Synthetizer," in *Proceedings of SIGGRAPH 85*, also published as Comput. Graphics, *19*, 3, (July 85), pp. 287-296.

Rank1863 Rankine, W. J. W., "On the Exact Form of Waves near the Surfaces of Deep Water," Phil. Trans. Roy. Soc., A 153, (1863), pp. 127-138.

Reev83 Reeves, W. T., "Particle Systems-A Technique for Modeling a Class of Fuzzy Objects," Transactions on Graphics, *2*, 2, (April 83), pp. 91-108.

Scha80 Schachter, B., "Long crested wave models," Computer Graphics and Image Processing, *12*, (1980), pp. 187-201.

Shep64 Shepard, D., "A two-dimensional interpolation function for irregularly spaced data," in *Proceedings 1964 ACM National Conference*, (1964), pp. 517-524.

TsBa86 Ts'o, P. Y. and Barsky, B. A., "Modeling and Rendering Waves: Wave-tracing using Beta-splines and Reflective and Refractive Texture Mapping", submitted for publication.