Vectorized Procedural Models for Natural Terrain: Waves and Islands in the Sunset

Nelson L. Max
Lawrence Livermore National Laboratory

Abstract

A ray-tracing procedural model is described, in which ocean waves and islands are rendered by different but related algorithms. The algorithms are based on analytic formulas involving arithmetic operations, trigonometric functions, and square roots, and are organized for a vectorizing compiler on a Cray 1, a "supercomputer" with a vector pipeline architecture. Height field methods are used, one vertical scan line at a time, to trace the direct rays to the ocean, where they are reflected. Approximate methods are then applied to find whether the reflected rays meet any other object on their way to the sky. The output, at eight bits per pixel, gives information for shading, e.g. the angle of the surface normal for rays meeting the islands, or the angle of elevation from the horizon for rays continuing unobstructed to the sky.

The output is recorded on a magnetic tape for each frame in one cycle of the wave motion, and plotted offline on a Dicomed D-48 color film recorder. The eight bits per pixel are interpreted by a color translation table, which is gradually changed as the wave cycle is repeated to simulate the changing illumination during sunset.

Key Words

Ray tracing, vectorized, pipeline, height field, natural terrain, color table animation, piercing, line buffer, reflection, procedural model, water waves.

C.R. Catagories 8.2, 3.14

Introduction

A ray tracing algorithm for raster computer graphics follows the ray from the observer through each pixel, as it meets and possibly reflects from surfaces in a scene. Newell [1],and Rubin and Whitted [2] have proposed procedural models for ray tracing. Each object comes with its own algorithm to detect whether and where a ray meets it, and to determine the reflection and/or shading. If algorithms requiring a search in a data base are excluded, and combinations of standard analytic functions are used instead, such algorithms can run efficiently on a vector pipeline machine. The algorithms reported here were implemented on a Cray 1, but the CDC Cyber 205 and the TI ASC have similar pipeline architectures. Such machines can perform arithmetic operations much more efficiently when the same operation is applied to large vectors of operands, in which case one says that the computation is vectorized.

This paper presents vectorized procedural algorithms for natural scenes, including reflections from water surfaces rippled by superimposed traveling sine waves. Minnaert [6] has excellent explanations for the optical phenomena during sunsets, and produced by reflections in water waves. Schlachter [3] has proposed a model for random wave fields, which involves table look-up of precomputed narrow band waveforms, and is thus not easily vectorized. Information International used an illumination model to produce a leader for Pyramid films [4], showing beautiful cycloidal waves in the sunset. More complex effects, such as the reflection of objects in the water, require the tracing of reflected rays. Whitted [5]describes an algorithm which exhaustively traces all reflected and refracted rays, but which operates very slowly. In the current work, each ray is reflected a maximum of two times from the water, and then continues toward the other objects in the scene, which are diffusely reflecting. This allows the tracing to be treated in a uniform vectorized manner, resulting in much higher efficiency.

Color Table Animation

Color table animation has been used by Shoup [7] on a single picture in a frame buffer. In the movie described here, this technique is used instead with a periodic cycle of frames to expand it to a longer film.

The pictures are plotted offline from magnetic tape, by a Varian V-75 minicomputer controlling a Dicomed D-48 color film recorder. The Dicomed operates at 8 bits per pixel, and can pass the 8 input bits through a color translation table before recording onto film. For the pictures presented here, the same data was input three times, and recorded once through each of the red, green, and blue filters, with a different color translation table in each pass. The 256 available colors in this "paint by numbers" scheme were divided up into groups; one group was used for green island colors, one group for brown island colors, and one group for sky or sun colors. Within each group, the numbers coded for illumination information which, together with the current position of the sun, was used to specify the color. The ray from the eye through the center of each pixel was traced until it ended up at its final destination, either piercing an island or continuing on to the sky, perhaps after reflecting once or twice on the water. A number in one of the groups was then assigned to the pixel, as described below. Note that this procedure makes anti-aliasing between islands and sky impossible.

To make the longest sequence in the film, one cycle of periodic motion was recorded on tape, and repeatedly rendered onto film. The sun was gradually moved around a great circle C in the sky, and the color tables were changed appropriately, so that the repeated frames were differently illuminated each time through the cycle.

Color Specification for Diffuse Reflection

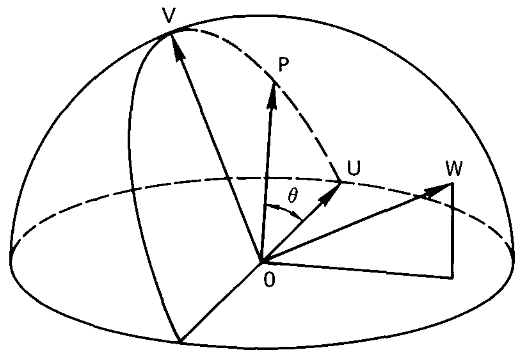

Figure 1

Figure 1 shows a unit sphere centered at the observer O, on which the circle C is projected. The vector U points to the setting sun, the vector V points toward the sun at its maximum elevation, and the vector W = U x V is normal to the plane of C. The angle θ between P and U is 2π/24 times the number of hours before sunset. The sun's position as a function of θ, is

$$P(\theta) = U \cos\theta + V \sin\theta$$

Suppose a ray traced from the observer through a given pixel ends up, perhaps after reflection, at a point on diffusely reflecting object where the unit surface normal is N. If N·P < 0, the surface faces away from the sun, and will be in shadow. If N·P > 0 the point may or may not be illuminated, depending on the presence of other surfaces between it and the sun. The pictures here were computed assuming no other surfaces cast shadows.

Since $N \cdot P = N \cdot U \cos\theta + N \cdot V \sin\theta$

$$= N \cdot U \sqrt{1-\sin^2} + N \cdot V \sin\theta,$$

N·P becomes zero when ρ(N) = sin θ, where

$$\rho(N) = -\frac{N \cdot U}{\sqrt{(N \cdot U)^2 + (N \cdot V)^2}} = -\frac{N \cdot U}{\sqrt{(1-N \cdot W)^2}}$$

For a given position P(θ) of the sun, the arc on the surface where ρ(N) = sin θ divides the surface into a region in light and a region in shadow. A sequence of these arcs divide the surface into regions which successively darken as the sun sets. Thus, a function of the form A + Bρ(N) can be used to specify the color number. When the sun is at position P(θ), those numbers which are greater than A + B sin θ correspond to shadows, while numbers decreasing from this value become brighter, with the number A + B sin (θ-π/2) the brightest. The two parameter family of normal vectors has been reduced to one parameter, which is not sufficient to simulate Lambert's law of diffuse reflection. Nevertheless, this shading scheme does indicate the shape of the surface.

In order to break up the boundaries between the "paint by numbers" regions, a random number between 0 and 1 is added to A + B ρ(N) before the result is truncated to an integer. The seed of the random number generator is set to the same value at the beginning of each frame, so the texture generated by this process will not jitter from frame to frame.

Color Specification for the Sky

The sky is normally a darker and more saturated blue near the zenith than near the horizon. During sunset, the brilliant sky colors also occur in approximately horizontal bands. Thus the z component of a ray reaching the sky can be used in computing a code for the color.

All the ray tracing and hidden surface computations are done by the Cray 1, and the Varian V-75 controlling the film recorder has no knowledge of the scene which produced a given raster image. However, the V-75 can compute a simple function of the color code in the input data and the raster position of the pixel to determine the output color on the film. The color translation is such a function, depending only on the input data. Another such calculation adds the sun's disk to the appropriate pixels by the rule: "If the pixel is above the horizon and entirely inside the sun's disk, and the input data specifies a sky color, then render the pixel sun color." An additional rule for anti-aliasing might read: "If the pixel is above the horizon and near the boundary of the sun's disk, and the input data specifies a sky color, render the pixel an appropriately weighted average of sun color and the sky color."

In order to show the glimmer of the sun reflecting in the water, some of the 256 possible numbers are assigned to identify the highlights produced by different possible sun positions. A ray reflected in the direction of the unit vector R can meet the sun's disk only if |W·R| < cos δ, where δ is the angular radius of the sun, about .75 , and W is the unit vector normal to the plane of the sun's motion, as in figure 1. If this is the case, then for afternoon positions of the sun, between U and V in figure 1, the z component R_z of R can be used to identify the time of day when these reflections produce the highlights. For other times of day, the same z component can be used to determine the sky color as before. Thus, a second range of z component codes is used to identify reflected rays along the track of the sun. As the unit vector P in the direction of the sun approaches a position with a specific z value, the corresponding table entry is changed smoothly from its sky value to the sun color, and then back again as the vector P passes and a new table entry brightens. At least one entry is sun color at any time, and adjacent entires are intermediate between sky and sun color. In addition, a third range of z component codes is used for values of $|W \cdot R|$ between cos δ and cos (3 δ), to give further intermediate colors for rays deviating horizontally from the sun's position. These intermediate colors allow the reflections to change continuously with the time of day, and serve to increase the area in highlight and to anti-alias the edge of the bright highlight areas. An appropriately scaled random number is added to $|W \cdot R|$ and to R_z before they are thresholded to determine the color. This again breaks up the boundaries between colored regions in the sky, and in its reflection. Figure 4 shows the result when the water is perfectly still. The sun's reflection is oblong because δ was chosen larger to brighten the glimmer, and is trapezoidal because the sun is following a diagonal course as in figure 1, so that W is not horizontal.

Water Waves

Waves on the surface of water are affected by two forces: gravity and surface tension. However, when the wavelength is longer than a few inches, gravity is the dominant force. The motion of the water is described by non-linear partial differential equations. In an approximate solution to these equations, valid for waves of small amplitude, the velocity c is proportional to the square root of the wavelength λ:

$$c = \sqrt{\frac{g\lambda}{2\pi}} = \sqrt{\frac{g}{k}}$$

Here $k = 2\pi/\lambda$ is called the wave number, and g is the acceleration of gravity. The height of the free surface can be approximated to first order by the wave train

$$z(x,t) = a \cos k(x - ct) = a \cos (kx - \omega t)$$

Here a is the amplitude, and $\omega = kc$ is the angular frequency in radians per second. On a two dimensional surface, the wave train is further specified by a wave vector (ℓ,m) such that $\ell^2 + m^2 = k^2$. The equation for the surface of a long crested wave train then becomes

$$z(x,y,t) = -h + a \cos (\ell x + my - \omega t)$$

where h is the distance of the mean sea level below the eye at z = o.

In the linear first approximation, these wave trains pass through each other without modification, and a wavey surface can be represented as a sum of several wave trains:

$$f(x,y,t) = -h + \sum_{j=1}^{n} a_j \cos (\ell_j x + m_j y - \omega_j t) \quad \ldots 1)$$

If a cycle of waves is to be repeated in a film, this function must be periodic in time, and the frequencies ω_j must all be multiples of some fundamental frequency. Since $\omega = kc$ and $c = \sqrt{\frac{g}{k}}$, this means that the wave numbers must be proportional to the square of the frequencies.

If the largest wave has a period equal to the repeating cycle, and wavelength λ, the available wavelengths are thus λ, $\lambda/4$, $\lambda/9$, $\lambda/16$, ... This puts some restrictions on the superimposed wave trains, but it is still possible to produce a reasonable looking surface. More random surfaces have been studied by Schachter [3] and Longuet-Higgins [8] .

Water waves with large amplitude have wide shallow troughs, and narrow more highly curved crests, as shown in figure 2. Thus the cosine wave approximation is inadequate for large amplitudes, and a more precise solution is required. There is an exact solution to the equations of motion in which a vertical section of the free surface takes the form of a cycloid.

If the water is excited from rest by wind on its surface, or a group of waves entering from a distance, one can prove that the curl of the velocity vector field remains zero, i.e., the flow remains irrotational. The exact solution just mentioned is not irrotational, and is therefore unreasonable.

Stokes [9] gave a method for calculating periodic irrotational waves of finite amplitude by successive approximation of their fourier series. If the waves are symmetrical about their peaks at x = 0, the fourier series takes the form

$$z(x,t) = \sum_{n=1}^{\infty} a_n \cos n k (x - ct) \quad \ldots 2)$$

Let $a = ka_1$ be the maximum slope of the first term of this series. Stokes solved for the other amplitudes as expansions in powers of a, and found, to fourth order,

$$a_1 = a/k$$
$$a_2 = (a^2/2 + 17a^4/24)/k$$
$$a_3 = (3a^3/8)/k$$
$$a_4 = (a^4/3)/k$$

Figure 2 shows the contribution of these first four terms to the shape of the wave. Schwartz [10]

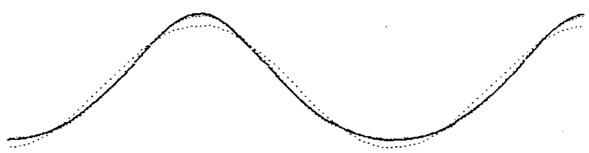

Figure 2. Stokes' approximations to water wave forms.

has recently found many more terms of this series by computer. The resulting waves are called Stokes waves, and form a one parameter family of shapes, depending on the value a. Their velocity also depends on a, but only in the quadratic and higher terms. This effect has been neglected here.

A few terms of the series 2) can easily be added into the sum 1) to give the waves of large slope a more realistic appearance. In the pictures here, only the second order term was added to the largest wave. For more speed, it would be possible to expand the first few terms of 2) as a polynomial in cos k (x - ct), so that the cosine need only be computed once.

Hidden Surfaces for Height Fields

Fishman and Schachter [11] have described an algorithm for rendering raster images of height fields, i.e., single valued functions of two variables. Assume f(x,y) is such a function, and an observer at (0,0,0) looks along the y axis at the surface $z = f(x,y)$, projected onto a picture plane at y=1. A vertical scan line in this plane at x=s contains the projections of points lying in the plane x=sy.

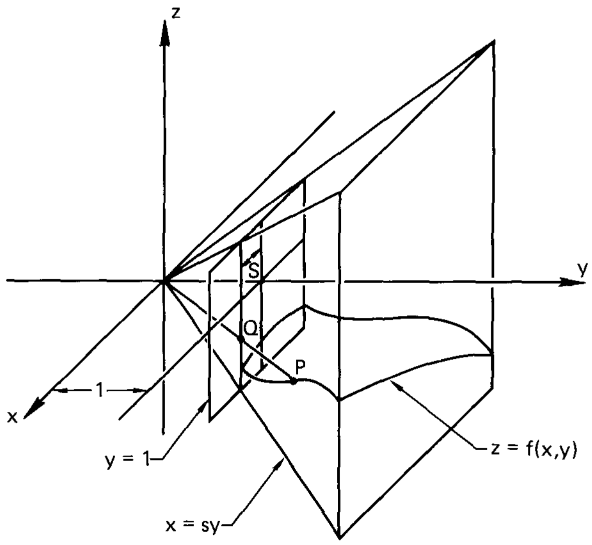

Figure 3. Perspective projection along a vertical scan line.

In figure 3 the point P = (sy, y, f (sy,y)) on the surface projects to the point Q on the picture plane at height $z = g(y) = f(sy,y)/y$. Although f is single valued, the function g(y) in general is not. The height field algorithm in [11] computes $g(y_i)$ for evenly spaced values of y_i, and assigns to a pixel the color of the surface at the first (sy_i, y_i) where $g(y_i)$ exceeds the z value for the center of the pixel. Such evenly spaced y_i are clearly inefficient for showing an unbounded surface in perspective, since the projected points become increasingly dense near the horizon. Instead, the expected density σ of projected values $g(y_i)$ per pixel should be constant. This will be the case if $y_i = h/(i\sigma)$, where the surface is assumed to lie near the plane $z = -h$.

This choice of y_i reproduces the detail of the small ripples near the eye, but does not waste time near the horizon. For the pictures here, $\sigma = 4$.

The values $g(y_i)$ are computed from equation 1) in a vectorized loop. In FORTRAN array notation, let $y(i) = y_i$ and $g(i) = g(y_i)$, and let h(k) be the z value in the picture plane of the center of pixel k. The following loop computes the array yp, where yp(k) is the approximate y value for the point projecting into pixel k.

```
      k = 1
      DO   30   i = 2,n
 40   IF (g(i).GT. h(k)) THEN
         yp(k) = y(i-1) + (y(i) - y (i-1))*
     1           (h(k) - g(i-1)) / (g(i) - g(i-1))
         k = k + 1
         GO TO 40
         ENDIF
 30   CONTINUE
```

This loop cannot be vectorized because the indices k change in an unpredictable manner, but it constitutes only a small part of the computation.

Once the values $y_k = yp(k)$ have been found, the partial derivatives $\partial f/\partial x$ and $\partial f/\partial y$ of the surface at each point $(x,y) = (sy_k, y_k)$ can be computed from equation 1):

$$\partial f\,(sy_k,\ y_k,\ t)\ /\partial x = -\sum_{j=1}^{n} \ell_j\, a_j \sin\,(b_j y_k + c_j)$$

$$\partial f\,(sy_k,\ y_k,\ t)\ /\partial y = -\sum_{j=1}^{n} m_j a_j \sin\,(b_j y_k + c_j)$$

where $b_j = \ell_j s + m_j$, and $c_j = -\omega_j t$

The range of k is much larger than the range of j, so the loop on k is made the inner one. It can be easily vectorized, taking advantage of the common subexpression $a_j \sin\,(b_j y_k + c_j)$. The surface normal $(\bar{x}, \bar{y}, \bar{z})$ is then computed by

$$\bar{z} = 1/\sqrt{(\partial f/\partial x)^2 + (\partial f/\partial y)^2 + 1}$$

$$\bar{x} = -\bar{z} \cdot (\partial f/\partial x)$$

$$\bar{y} = -\bar{z} \cdot (\partial f/\partial y)$$

If a ray is traced from the eye to the point (sy, y, f (sy, y)), its reflection can be found using the normal vector $(\bar{x}, \bar{y}, \bar{z})$ and the formulas in Blinn [12]. The shading will be computed by tracing this reflected ray to its final destination.

The Islands

The same height field hidden surface algorithm is used for the islands, which are represented as elliptical paraboloids with superimposed cosine terms to give rolling hills. However, the shading is determined from the direction of the normal vector, rather than from the reflected ray.

The elliptical paraboloid for island k is given by

$$P_k(x,y) = h_k - e_k(x-X_k)^2 - f_k(x-X_k)(y-Y_k) - g_k(y-Y_k)^2 \qquad \ldots 3)$$

where (X_k, Y_k, h_k) is the highest point on the paraboloid. After the cosine terms are added, the surface becomes

$$Q_k(x,y) = P_k(x,y) + \sum_{j=1}^{n_k} a_{jk} \cos(\ell_{jk}x + m_{jk}y) \qquad \ldots 4)$$

The maximum perturbation that the cosine terms can make is

$$d_k = \sum_{j=1}^{n_k} a_{jk}$$

Similarly, the maximum perturbation a water wave from equation 1) can make on the sea level -h is

$$d = \sum_{j=1}^{n} a_j$$

Therefore, any visible points (x,y) where the island protrudes above the level of the ocean lie inside the ellipse

$$P_k(x,y) + d_k + h + d = 0$$

The sample values (sy_i, y_i) need only be taken inside this ellipse.

The height field computation is performed for the ocean surface and also for all the islands, and the surface closest to the viewer is chosen to shade each pixel. This gives a brute force depth buffer hidden surface algorithm on each vertical scan line, which gains its efficiency through the vector pipeline processing.

Each island is actually formed from three paraboloids of different colors. The beach is a shallow brown paraboloid, which is rendered tan because of its nearly horizontal surface. The cliffs are rendered by a steeper brown paraboloid, and the rolling hills above them are made of a third green paraboloid. The simple depth buffer algorithm could take the union of these three paraboloids and compute the nearest point. However, when comparing the hills above the cliffs, it is actually the farther of these two surfaces which is visible, so special logic is required. This is simplified by defining the hills by the same equation as the cliffs, and taking the top of the cliffs to lie in a single plane. Whenever the equation generates a point above this plane, the height above the cliff level is decreased by a constant factor, and the point is colored green. A more realistic cliff line would result if a separate equation were used for the hills.

Tracing Reflected Rays

A ray reflected from the ocean must be traced to find out if it pierces any other objects before meeting the sky. Since these rays do not originate at the eye, an involved computation would be required to reproduce for reflected rays the accuracy provided by the height field algorithm for the direct rays. Instead, the scene is approximated by simpler quadratic surfaces, which permit direct ray piercing algorithms.

For the purposes of ray piercing, the islands are represented by equation 3), without the cosine terms. A position S at a distance t along the ray in direction (R_x, R_y, R_z) from the point (Q_x, Q_y, Q_z) is given by the vector function

$$(S_x,S_y,S_z) = (Q_x+tR_x, Q_y+tR_y, Q_z+tR_z) \qquad \ldots 5)$$

To find the distances t_1 and t_2 at which this ray pierces the paraboloid, S_x and S_y are substituted into equation 3) and the resulting surface height is set equal to S_z; giving

$$P_k(Q_x+tR_x, Q_y+tR_y) - Q_z - tR_z = 0$$

Since $P_k(x,y)$ is quadratic in x and y, this yields an equation in t of the form

$$at^2 + bt + c = 0 \qquad \ldots 6)$$

whose coefficients are expressed in terms of the constants in equations 3) and 5).

The well-known quadratic formula gives

$$t = \frac{-b \pm \sqrt{d}}{2a}$$

where the discriminant $d=b^2-4ac$. The quadratic formula is vectorized by using the absolute value of d in the square root, and the smallest root t_1 is found. If d is negative the ray actually misses the paraboloid, and if t_1 is negative, the ray meets it in the negative of the reflected direction. These cases are eliminated in a second vectorized loop, which also checks whether t_1 is less than the distance to the closest island found so far for the ray. If all the conditions are met, the name and distance to the closest island are updated.

Since the islands represent the intersections of cliff and hill paraboloids, the ray must pierce both paraboloids to hit the island. If the hill surface has the same equation as the cliff, except for an abrupt change in slope at the cliff edge, the formulas for the coefficients a, b, and c of equations 6) for the two paraboloids contain many common subexpressions, which leads to greater efficiency.

When the nearest island (if any) has been found for each ray, the surface normal at the piercing point is evaluated from equation 4) with all the cosine terms, and used to find the shading as before. This is similar to the approximation used by Blinn [13] to render wrinkled surfaces; the wrinkles show up in the interior shading, but not on the profile. The discrepancy in the reflection of the profile would never be noticed if there were ripples in the water. Figure 4 shows a reflection in completely still water. The bright highlight on the cliff results from normal vectors which face towards the sun, computed by the wrinkle algorithm at piercing points which would

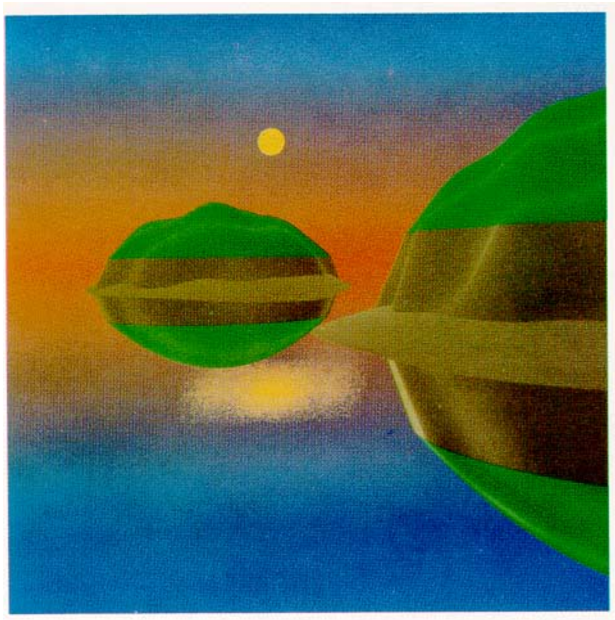

Figure 4. Islands near sunset in still water.

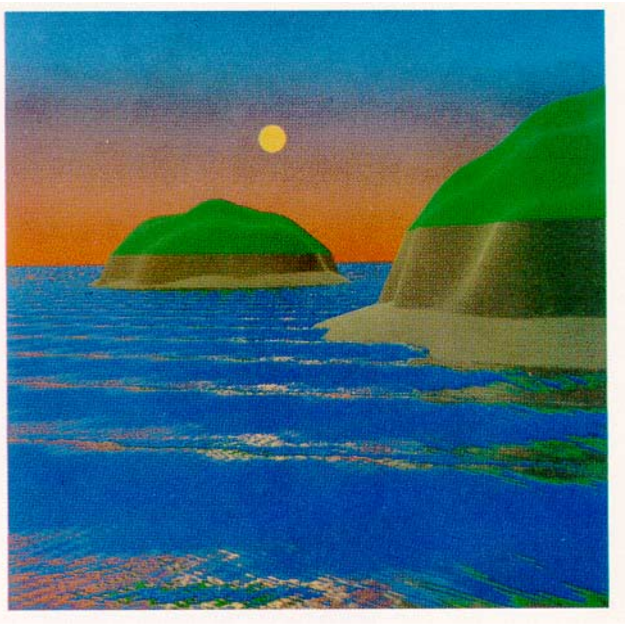

Figure 6. Islands near sunset allowing two reflections from waves.

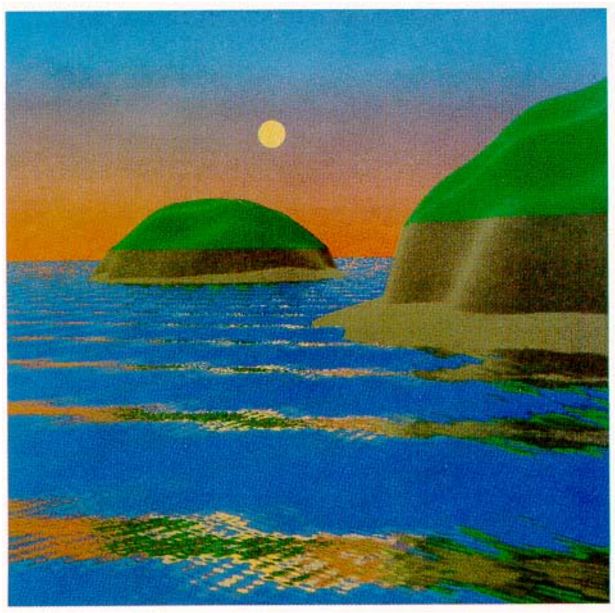

Figure 5. Islands near sunset with one reflection from waves.

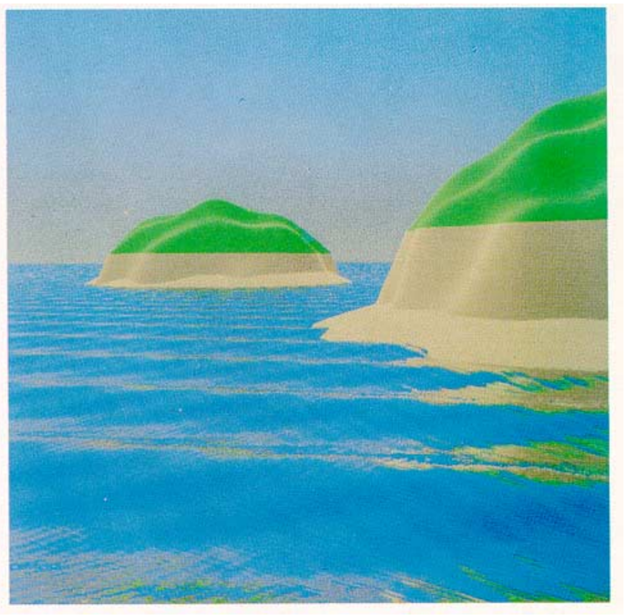

Figure 7. The same data as in figure 6, with early afternoon lighting.

not actually be visible to the observer if the wrinkles were taken into account.

It is also possible for a ray from the observer to be multiply reflected from the water surface. Rays glancing off the downslope of a wave which faces away from the observer reflect in a direction with negative z component, or with insufficient positive component to clear the next wave. Also, since the height function algorithm is only approximate, the interpolated value of y may correspond to a point Q on the surface which is actually hidden from the observer, and the ray from the observer may even approach Q from below the water surface. In figure 5 the z component of each reflected ray has been replaced by its absolute value, and no secondary reflections have been processed. The colorful bands on the waves' downslopes are produced by rays which in reality would never reach their simulated final destination. For a more realistic (but less colorful) image, secondary reflections on the water must be taken into account.

For this purpose, the first crest of the wave of largest amplitude in front of the reflected ray is approximated by a parabolic cylinder which is tangent to the crest and has the same curvature as the Stokes wave at the crest. A parabolic cylinder is a degenerate form of an elliptical paraboloid, and a simplified version of the piercing algorithm discussed above is used to find the piercing point. The normal to the actual wave at the x, y coordinates of the piercing point is then used to find the second reflected ray. It is still possible that a third reflection could occur, but this is unlikely, and is not considered. Instead the z component of the second reflected ray is replaced by its absolute value.

About 10% to 15% of the rays undergo secondary reflection. When these cases are detected from the discriminant of the quadratic equation 6), their data is copied ("compressed" or "gathered") into consecutive positions in auxilary data vectors. This allows subsequent calculations for these cases to be efficiently vectorized. The resulting second reflected ray is then copied back ("decompressed" or "scattered") into the arrays containing the first reflected rays, so that all the rays can be continued together to meet the islands or the sky.

Figures 6 and 7 show the results of this algorithm. They are made at 1020 x 1023 pixel resolution, using the same 8 bit data, with different color translation tables. The computation time was 34 seconds on the Cray 1, and the plotting time 5 minutes on the Dicomed, including 45 seconds to backspace the tape twice between color passes.

Directions for the Future

There are several possible improvements for these pictures. First, no account has been taken of refraction. According to Fresnel's law, (see Cook and Torrence [14]) a glancing ray is almost completely reflected by a water surface, but for rays closer to the surface normal, a substantial fraction of the energy is refracted into the water. Conversely a ray approaching the surface of the water from below will have this same fraction refracted into the air. Sunlight scattered back up by small particles in the water may refract toward the observer. Thus the front surfaces of waves take on the color of the water rather than the color of the sky, and appear dark green. To handle this refraction correctly, one would need many interpolated colors between sky color and water color, and initial tests indicated that an 8 bit color table is inadequate for this purpose. A larger (software) color table could be used, or the ray tracing program could output the actual red, green, and blue values for each pixel instead of one color table index.

When sunlight reflects from rippled water and then strikes another object, one can see ripples of light on the object. It should be possible to trace a family of sun rays meeting a small patch of water, and then trace the reflected and refracted rays until they pierce a small object extending above and below the water line. The rays could be averaged into an array representing the surface of the object, to create the ripples of illumination. The illuminated object could also be seen by reflection in the water, and by refraction through it, modified and attenuated by the scattering.

Finally, it should be possible to render clouds by the height function techniques. The deviations of the clouds from a mean cloud level could be computed using formulas involving polynomials, cosines square roots, ...etc. The shading would not depend on the normal vector, but rather on the depth below mean cloud level. The height field algorithm could be modified to handle double valued functions whose domain is only a subset of the plane. As before, an approximation would replace the height field algorithm for reflected rays. These would be traced till they pierced the mean cloud level. If the piercing point lay in the domain of the cloud function, the shade would be assigned according to the color of the cloud below the x, y coordinates of the piercing point. Clouds between the sun and the water would intercept the rays generating the highlights, and cause shadows in the sun's glimmer on the water. The clouds could also glow in the sunset.

Acknowledgments

This work was inspired by the films of Turner Whitted and Loren Carpenter. I would like to thank Turner Whitted for encouraging conversations, John Engle, Al Shannon, and Frank McMahon for helping me vectorize the FORTRAN code for the Cray 1, John Guckenheimer for pointing out references [9], the reviewers for pointing out references [3] and[11], Don Faul and Steven Williams for carefully reading the manuscript, Dick Rau and Rick Simms for making last minute color corrected prints, and many friends and colleagues for suggesting ways to make the pictures more realistic. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract W-7405-Eng-48.

Disclaimer

References

[1] Newell, M. The Utilization of Procedure Models in Digital Image Synthesis. PhD Thesis, University of Utah (1975)

[2] Rubin, S., and Whitted, T., A 3-Dimensional Representation for Fast Rendering of Complex Scenes. Computer Graphics Vol. 14, No. 3 (1980) pp. 110-116

[3] Schachter, B., Long Crested Wave Models. Computer Graphics and Image Processing Vol. 12 (1980) pp. 187-201

[4] Pyramid Catalogue, (1981) Pyramid, Box 1048 Santa Monica, or poster available from Jannes Art Publishing, Chicago

[5] Whitted, T., An Improved Illumination Model for Shaded Display. Comm. ACM Vol. 23, No. 6, (1980) pp. 343-349

[6] Minnaert, M., Light and Color in the Open Air. Dover Publications, Inc. (1954) New York

[7] Shoup, R., Color Table Animation. Computer Graphics Vol. 13, No. 2 (1979) pp. 286-292

[8] Longuet-Higgins, M., The Statistical Analysis of a Random, Moving Surface. Proc. Roy. Soc. of London Vol. 249 (1957) pp. 321-387

[9] Stokes, G.G., Mathematical and Physical Papers. Vol. 1 (1880) p. 341, Cambridge University Press

[10] Schwartz, L., Computer Extension and Analytic Continuation of Stokes' Expansion of Gravity Waves. J. Fluid Mech. Vol. 62, part 3 (1974) pp. 553-578

[11] Fishman, B., and Schachter, B., Computer Display of Height Fields. Computers and Graphics Vol. 5 (1980) pp. 53-60

[12] Blinn, J., Models of Light Reflection for Computer Synthesized Pictures. Computer Graphics Vol. 11, No. 2 (1977) pp. 192-198

[13] Blinn, J., Simulation of Wrinkled Surfaces. Computer Graphics Vol. 12, No. 3 (1978) pp. 286-292

[14] Cook, R., and Torrance, K., Reflectance Models for Computer Graphics. Vol. 15, No. 13 (1981)