THE REPRESENTATION OF WATER

Geoff Wyvill
University of Otago Computer Science Department
Box 56, Dunedin, New Zealand

Andrew Pearce and Brian Wyvill
University of Calgary Computer Science Department
2500 University Drive N.W. Calgary, Alberta, Canada

Abstract

Water is the commonest of everyday substances and its many forms have provided subjects for artists for as long as art has existed. But in computer graphics, there seem to have been few attempts to make pictures of water. The reason for this is simple. Realistic pictures of water are very hard to produce. We examine some of the reasons for this difficulty and report on some of our own experiments.

Background

Water is all around us and plays a part in many natural scenes. But it rarely appears in computer graphics images. The last four years has seen an enormous increase in interest in modelling natural phenomena. Fournier [1982] and others have produced realistic terrains using approximations to fractal surfaces building on the ideas of Mandelbrot [1983]. Reeves [1983] has modelled fire and Gardner [1985] has made impressive clouds. Although Reeves suggests that his particle systems can be used to model water, his paper gives no example. Perlin [1985] has published a picture, *Ocean Sunset,* showing a representation of a seascape and this is probably the best representation to date. However, it shows only one appearance of water and it is not clear how the method should be generalised. Water reflections appear in *Road to Point Reyes* [Cook 1983] but they are very simple, as is the pool with ripples in *Erehwon* [Weliky 1985]. Nelson Max [1981] made some pictures of a pair of islands in the sunset and his discussion deals with some of the problems mentioned below. It is, unfortunately, difficult to assess Max's ideas from the pictures themselves as he was hampered by using a display with only 256 colours.

Water is difficult to represent for several reasons. Most of the water we see is in motion. Its shape depends on this motion and the motion is very complicated. A full, hydrodynamic simulation can, in principle, provide us with a complete description of the shape of any mass of water, but the computational requirements would be huge.

Even when we know the shape of a mass of water, it is still difficult to render because of its optical properties. Most of the light falling on it is refracted or reflected, but even light which passes through the water gets scattered in a more or less complex fashion. And the appearance is further complicated by the fact that any surface below the water is illuminated indirectly by rays focused and scattered by refraction at the surface. Water in lakes, ponds and puddles presents the simplest surface, a plane disturbed by combinations of waves. The wave shapes are affected by varying depth and boundary. Water flowing in streams and rivers is far more complicated. We have initiated a research project aimed at studying all aspects of the appearance of water. In particular we are experimenting with the technique of *soft* objects [Wyvill 1986] to provide a general model for the more complicated cases: streams, waterfalls and fountains.

In this paper, we have confined ourselves to the study of pools of water with waves. We have not yet attempted animation, and we have avoided actual physical simulation. Our main purpose is to discover which features matter most when presenting a realistic picture of water with waves. Another way of looking at it is to ask which features can be omitted without making the picture *unconvincing.* Our standard of assessment is thus rather subjective. Still photographs of moving water often look very different from the original because we never observe waves over an area at the same moment. Ideally we should be trying to compare our artificial pictures with photographs of water in similar circumstances. This we have not done.

This work has been conducted as part of the Graphicsland computer animation project at the University of Calgary [Wyvill 1985]. The Graphicsland system provides a set of programs which are used as tools for creating pictures and animations using three dimensional, computer models. The pictures we have created, all use a ray tracing program which is part of the Graphicsland system. This program has been modified, however, to support some of our special techniques.

Relevant Features

The appearance of water in ponds, pools and puddles varies enormously. Sometimes the water is opaque (or very nearly so) because of mud or other particles and sometimes it is transparent. There is a *range* of colours which are convincing because they do occur in nature whereas other combinations evidently do not. To investigate all this systematically is a huge undertaking but we have made a start by examining those features in earlier work which seem less satisfactory.

Wave Shapes Waves on water are complicated even in the restricted case of pools. The simplest component is a single wave front resulting from some disturbance at one point. The resulting wave consists of an up-and-down motion of water which spreads in all directions from the centre of the disturbance. At any point, the motion can be viewed as growing quickly in amplitude as the wave front first arrives and then slowly decaying. In practice, such a single wave is almost never seen. Wind and other forces continuously make disturbances at many points on the surface and any travelling wave produces secondary, reflected waves wherever it meets the edge of the water.

Nelson Max [1981] went to some trouble to find out what was a reasonable shape for a water wave profile. It seemed to us that the shape produced by just two intersecting waves was so complicated that one would not expect to detect a 'wrong' shape just by eye. Yet Max's sea waves look very wrong in the vicinity of his islands. The reason for this is that the waves do not show any radical change of appearance as they approach the islands. The sloping beach simply intersects what looks like a deep-sea wave. In the presence of an important optical cue such as the beach, we have expectations which must be met if the picture is to look convincing. In *Ocean Sunset* Perlin [1985], we see a remarkably convincing picture of waves on the ocean. Because there are no other objects to give us impressions of scale or expectations about the waves, we are easily convinced. Can we make adequate pictures of water in a context where we have expectations about the size and behaviour of the waves?

To produce realistic wave shapes on a sloping beach is very difficult so we have chosen to model waves in a rectangular pool with vertical sides. We felt that it was better to use *some* context (unlike Perlin) but we needed to keep it simple enough that we could represent the boundary without recourse to detailed physical simulation.

Max [1981] uses combinations of linear wave fronts. Perlin [1985] observing that these were too regular, combined spherical wave fronts from a collection of point sources whose positions were randomised. Our best results have also been obtained by adding waves from point sources. In some cases we have used a few sources placed at random, in others we have tried to use some knowledge about the scene to choose the source positions. The waves in Figure 5 are generated from a single main source together with eight 'secondary' sources. These secondary sources have been so placed that they represent *reflected images* of the primary source in the sides of the pool. This means that where the waves meet the sides of the pool, their shape is consistent with the expectation that any wave meets a reflection of itself at the edge. Thus the waves of Figure 5 are, in a sense, characteristic of the waves in a rectangular swimming pool.

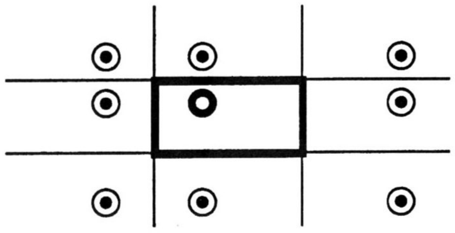

Figure 1

Wave source **O**, and images ⊙, in a pool

Interestingly enough, there is no decay in our waves. They are combinations of sine waves travelling in different directions. The vertical displacement at any point x,y in Figure 5 is:

$$\sum_i w_i \sin(kr_i)$$

Where r_i is the distance from x,y to one of the wave sources and w_i is the amplitude of that source. The reason why this works is not clear. But it is a matter of common observation that waves in a swimming pool are of pretty uniform size due to the mixing of many disturbances.

Colour Figure 8 shows blue, opaque water. Figure 9 shows green, partially transparent water. Neither is *correct.* What you regard as acceptable depends on your expectations. If the swimmer is in a swimming pool, Figure 8 is more realistic. If he is in a canal, you will probably prefer Figure 9. In no case have we used a very reasonable optical model for the water. The colour in water is a function of depth. Light entering the water is absorbed and scattered so that every part of the water below the surface behaves as a secondary light source and the intensity of these sources is a decreasing function of depth. Light from these sources is further absorbed and scattered, so the colour of a body of water is difficult to predict theoretically even if the shape is very simple and reflection and refraction are ignored. Again, the complexity of this acts in our favour in that we do not know what to expect. In all our pictures, the colour is produced by a simple mixture of light reflected and refracted at the surface. No further filtering or volume dependent effects are used.

Modelling technique Max [1981] uses a functional approximation to the shape of his waves and performs direct ray tracing. The ray intersections are found by iteration. Perlin [1985] is using a scan line algorithm and solid texture mapping. This method is very versatile and enables colour and texture to be changed by post processing the pixel buffer after the scene has been rendered. The method does not, however, support ray tracing so no reflections are possible.

We wanted to compare different approaches so we have used three different techniques. The first is to construct an approximation to our wavy surface using polygons. This is shown in Figure 5. Polygons are not very suitable, but the *Graphicsland* system provides an easy way to create and manipulate such models so this provided us with a convenient reference surface for no extra programming effort.

The second technique is to use our wave function as a texture map. This basically follows Blinn's classical method of bump mapping [Blinn 1978] except that we are ray tracing. The intersection of each ray with the plane water surface is found and then a false surface normal is computed for that intersection point. The direction of reflected and refracted rays is then found using this false normal. This technique is not new. If it has been described explicitly we are not aware of it, but it has certainly been used in published pictures, e.g: *Erehwon* [Weliky 1985]. Bump mapping works because we cannot perceive the three dimensional position of part of a distant surface very accurately. We deduce the finer detail of shape from light, shade and reflections from the surface. This is another reason why we do not think that the shape of the wave profile is very important. Indeed we do not know what shape our texture-mapped waves are supposed to be, only how the normals are perturbed.

Our third technique is a new application of displacement mapping [Cook 1984]. The idea of a displacement map in scan-line algorithms is to produce local variations of shape by calculating for each surface element a displaced position in space. Using this method, a block can be represented by just a few polygons, and given a curved, sculptured appearance as it is rendered. The advantage of the displacement map is that it enables us to represent the profile where the waves meet the pool side. This feature is missing in the texture-mapped examples.

Experimental Method

We have set up some standard scenes using the tools of *Graphicsland* and rendered them using a specially modified ray tracing program. This has enabled us to generate pictures showing a simulated water surface in which we can change relevant variables selectively. Thus in one picture we make the water opaque, in another transparent. We can use different patterns of surface wave and experiment with our three different rendering techniques.

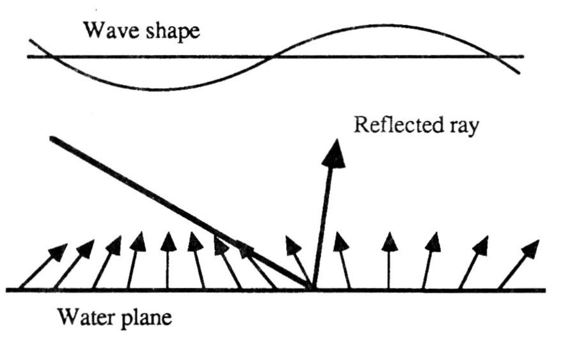

Figure 2 False normal vectors

The Ray Tracer

The principal enhancement of our ray tracer is the ability to recognise ray intersection with an object to which texture mapping applies. Thus the intersection with the water plane is given a surface normal which has been modified by our wave function. This is illustrated in Figure 2. Every point in the plane of the water surface has its normal vector modified by the wave function. The modifications calculated for all the wave sources are added as vectors to give a normal vector which can deviate from the vertical by a small amount in any direction. This calculation is very fast, but not quite the same as finding the true normal of the combined wave. At this stage, we do not know how important the difference is, but the reflections in Figures 8 and 9 seem good enough.

To produce a profile edge more cheaply than by constructing a genuinely wavy surface, we have also experimented with a displacement map. The idea is that we calculate an ordinary point of intersection with the plane surface which is our water. Then we modify the point of intersection, making it nearer or further from the eye, along the line of sight, according to our wave function. Having got a modified point of intersection we then check back against adjacent objects in case, after all, the point of intersection is now further from the eye than a point of intersection with some other object. For example, in Figure 3, the intersection of the ray and plane surface is in front of the pool's side. But the modified intersection is behind. In this case the ray tracer 'sees' the pool's side, not the water. The effect is to produce an artificial profile edge, Figure 6.

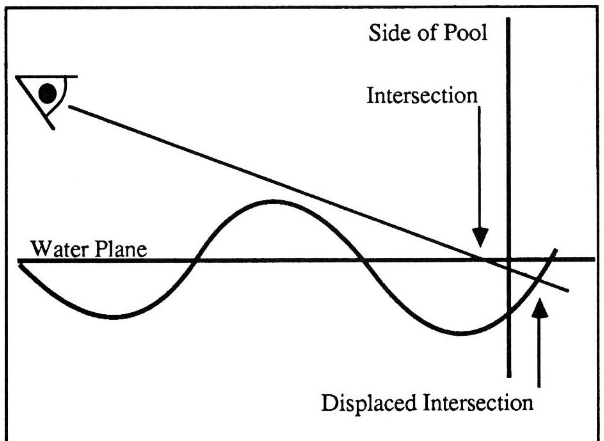

Figure 3 Displacement Mapping

Once again, the wave shape calculated by this displacement map is different from the original. What is worse, the effective shape of the wave depends on the angle of view. Our ray tracer uses a system of uniform space division to reduce the number of ray intersection calculations, and tests for adjacent objects are confined to the current volume of space division. For these reasons the method is less general than we would like, but it does offer a cheap way to represent the wave profile.

Max [1981] reports that in his scenes 10 to 15% o rays were reflected twice by the water surface. Our texture mapping and displacement mapping techniques do not allow for this possibility. We are not sure how important this is. Max shows two pictures rendered with and without this second reflection, but there are other differences in the two scenes which make it difficult to be sure how much this affects our impression of the water.

The photographs

We present a selection of images to illustrate our techniques. Figure 4 is an abstract scene using texture mapping for the water surface. Figure 5 shows the same scene with the polygonal representation. Displacement mapping is used in Figure 6 and the water colour and reflectivity have been changed in Figure 7.

Figures 8 and 9 show the effect of tuning the variables in an attempt at realism. The water in Figure 9 is transparent: note the distorted lower jaw. Our swimmer has no body beneath the head so this picture is a little inconsistent.

Conclusion

We have conducted a series of experiments to determine how to make convincing pictures of water in pools and puddles. We have concentrated on using a simplified model of waves on the surface and a variety of 'tricks' to achieve reasonably fast rendering.

By careful choice of reflectance, colour and other properties, we can make quite convincing pictures of water in this way. But there are many more avenues to be explored. So far, we have made no attempt to simulate the criss cross patterns of light due to refraction and focusing of light sources. This is an important feature of the appearance of shallow water.

Acknowledgement

This research has been partly funded by the Natural Sciences and Engineering Research Council of Canada.

References:

James F. Blinn 1978
Simulation of Wrinkled Surfaces
Computer Graphics Vol 12 No 3 pp 286-292 August

Robert L. Cook, Loren Carpenter, Thomas Porter, William Reeves, David Salesin and Alvy Ray Smith 1983
Road to Point Reyes
Computer Graphics Vol 17 No 3 title page picture, July

Robert L. Cook 1984
Shade Trees
Computer Graphics Vol 18 No 3 pp 223-232

Alain Fournier, Don Fussell and Lorin Carpenter
Computer Rendering of Stochastic Models
CACM Vol 25 No 6 pp 371-384

Geoffrey Y. Gardner 1985
Visual Simulation of Clouds
Computer Graphics Vol 19 No 3 pp 297-303

S. Haruyama and Brian Barsky 1984
Using Stochastic Modeling for Texture Generation.
IEEE Computer Graphics and Applications pp 7-19 March

Benoit Mandelbrot 1983
The Fractal Geometry of Nature
W.H. Freeman and Co

Ken Perlin 1985
An Image Synthesizer
Computer Graphics Vol 19 No 3 pp 287-296

William T. Reeves 1983
Particle Systems — A Technique for Modeling a Class of Fuzzy Objects
Computer Graphics Volume 17 No 3 pp 359-376

Michael Weliky 1985
Erehwon
Computer Graphics Vol 19 No 3 cover picture, July

Brian Wyvill, Craig McPheeters and Rick Garbutt 1985
A Practical 3D Computer Animation System
The BKSTS Journal (British Kinematograph Sound and Television Society), 67 (6) pp 328-332, July

Geoff Wyvill, Craig McPheeters and Brian Wyvill 1986
Soft Objects
Proceedings of CG Tokyo, to be published

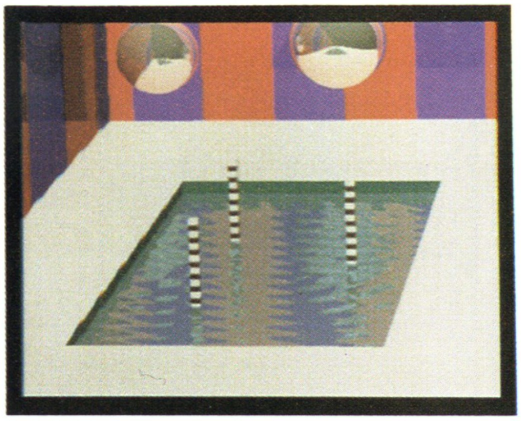
Figure 4 Abstract scene

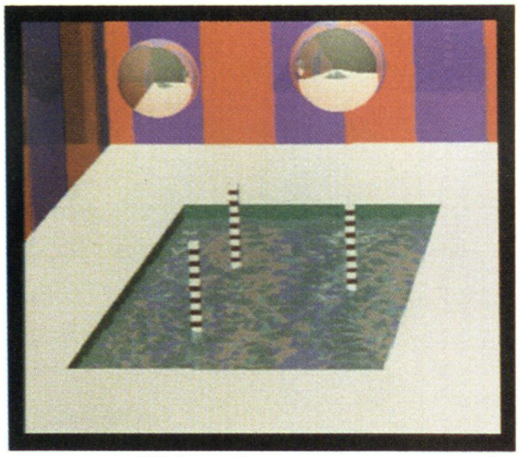
Figure 5 Polygon water

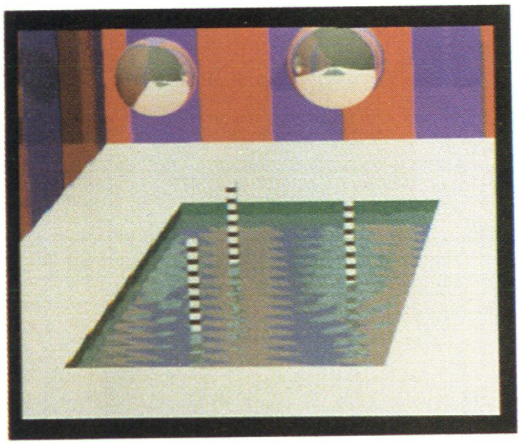
Figure 6 Displacement map

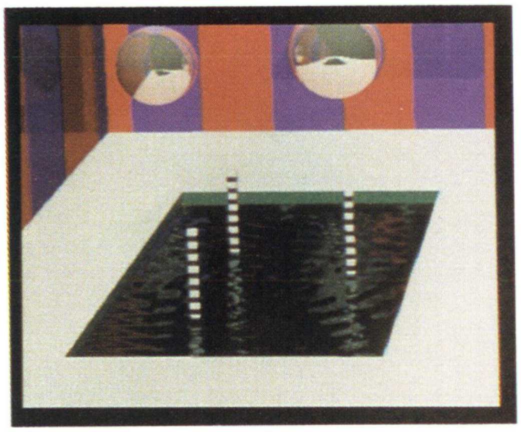
Figure 7 Effect of colour changes

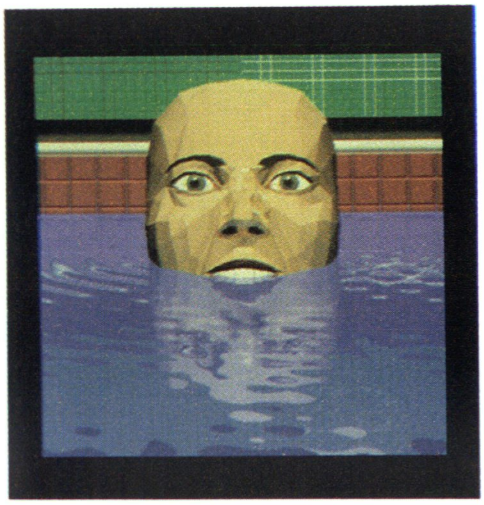
Figure 8 Swimmer

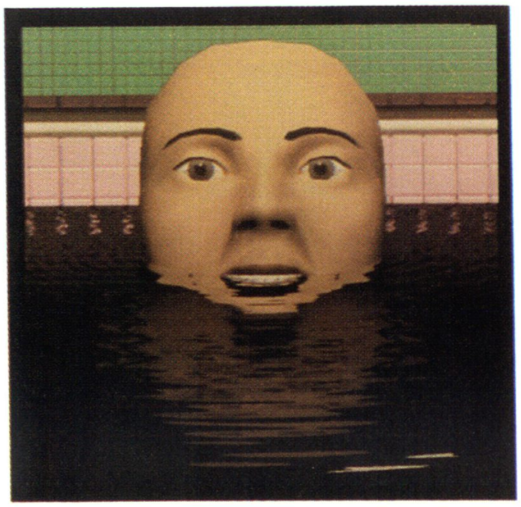
Figure 9 Effect of transparency