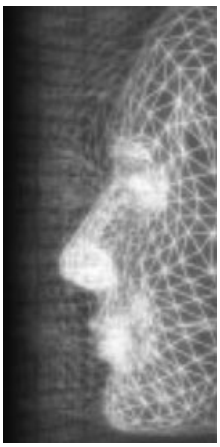
COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2008; **19**: 375–385
Published online in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/cav.239

Modeling the interaction between objects and cartoon water



By Jinhui Yu, Jing Liao and John Patterson*

In this paper, we describe a method for modeling the interaction between objects and the cartoon water which does not involve physics simulations. Our method involves the definition of flow path lines in the presence of obstacles, modeling of different types of water forms, and combinations of them in space and time. There are several notable features with our method: easy setting with little user intervention, modeling of complex water forms that represent more energetic water behavior than has been encompassed by semi-automatic means so far, and the adaptive change of animated water forms to the variation of obstacle object in number, size, and shape in the water path. A number of formulae for managing shape and time variance in these animations are given. We tested our method with both still and moving objects that have different shape and size in the water and relevant examples are given in the paper. Copyright © 2008 John Wiley & Sons, Ltd.

Received: 25 June 2008; Accepted: 25 June 2008

KEY WORDS: cartoon animation; effects; water; model

Introduction

Water exhibits different behavior under various conditions. Variables include, flow rate, interactions between barriers, rocks and other underwater obstacles, wind conditions, etc. Animated cartoon water effects aim to simulate this environment, albeit stylistically. Where water animation is done wholly by drawing, animators cannot avoid having to draw *different* water series if any of the conditions mentioned above change by themselves or in combination, for instance, obstacle objects vary in shape, size, and orientation of the movement, and this involves a lot of manual work.[1]

Drawn water animation can often expose problems which seem easier to solve by making 3D models and rendering them nonphotorealistically. However, 3D cartoon water model making has its own difficulties in respect of how the animator might stylize appearance and behaviors (visual abstraction and motion abstraction[2]) under various conditions. Some researchers are considering the question of what is the least amount of 3D geometric information you can get away with when trying to render out drawn animation which has the same liveliness and energy as a wholly drawn sequence. One way of doing this is to use procedures for generating or modifying the appearance of scene components but this is a very lightly explored area and there is no consensus about the nature of such procedures.

This paper discusses a 2D procedural approach to the animation of certain water effects such as those caused by stones in a running stream, or the animation of boats and animals moving in the water. Our goal is to animate cartoon water so that it interacts with both still and moving objects with little user intervention. To achieve this goal there are three main research challenges:

(1) Construction of models which capture the features of hand drawn cartoon water forms, including forms representing water waves and their interaction with objects, and forms depicting water flowing round objects.
(2) Adaptation of flow lines around obstacles when these are introduced into the water stream, and the adaptive deformation of different water forms as the water depicted flows over and around still objects or flows around moving objects (like boat hulls).
(3) Combination of different water forms in space (appearance) and time (behavior); this involves the effect of underlying structures on surface appearance, also controlling mechanisms for water

*Correspondence to: J. Liao, State Key Laboratory of CAD & CG, Zhejiang University, Hangzhou 310027, China.
E-mail: liaojing@cad.zju.edu.cn

forms with varying styles of temporal behavior, point-of-interaction detection between objects and water forms and its effect on the body of the water.

We address these issues by modeling selected cartoon water form primitives, grouping them according to rule categories in space and animating them in the path grid in time. The key idea in our method is the use of a path grid which allows changes in the flow path lines, deformation of water forms and point-of-interaction detection to link objects forming obstacles to water flow to the cartoon water forms. In our method, we require users to specify only a few control points to indicate the path of the water flow and the shape of the boundary between objects and the water surface. The system then generates the required water forms and behavior. A significant advantage of our method is that animated water forms can be adapted to the variations of obstacles in number, size, and shape. We have tested our method with both still and moving objects of varying shape and size placed in the water flow and examples are given later in the paper.

Related Work

During the last two decades, researchers have devoted much attention to the animation of water moving under various conditions. Detailed reviews of techniques for realistic water modeling and rendering have been presented by Iglesias[3] and Adabala and Manohar.[4] Our focus is on cartoon effects animation, where relatively little work has gone into this area. In fact existing work on modeling cartoon effects has focused only on two categories: gaseous phenomena (smoke, clouds) and some water phenomena.

In regard to gaseous phenomena, an early model for cartoon fire generation was proposed by Yu and Patterson,[5] in which quadratic curves (forming the flame boundaries) are fitted onto a fan-like structure and animated by alternating the positional parameters. Selle *et al*.[6] generated cartoon style animation of smoke by using a physically based simulator to drive particles rendered using a variant of the depth differences technique. A system for stylized animation of gaseous phenomena was presented by Di Fiore *et al*.,[7] here hand-drawn flames, drops, smoke puffs, and speed lines are animated along the 3D trajectory. McGuire and Fein[8,9] rendered amorphous shapes like smoke, clouds, and fluid with two-tone shading, self-shadowing, and silhouettes in a cartoon style. A model of smoke using a set of particles was proposed by Xu and He;[10] here physical parameters, such as velocity and force, are defined on the particles directly and the smoke is rendered with two-tone shading and silhouettes.

In regard to the animation of water phenomena, existing methods for cartoon water modeling can be subdivided into three categories:

(1) Particle based, such as the method by Thornton[11] in which a modular rig composed of a series of nodes is created to describe the shape of the splash, and particle emitters are constrained to these nodes to produce splash effects.
(2) Fluid simulation based, such as the work by Eden *et al*.,[12] this method takes as input a liquid surface obtained from a 3D physically based liquid simulation system, and stylizes this with bold outlines, patches of constant color to highlight near-silhouettes and shallow areas on the liquid surface.
(3) Template based, such as those proposed by Yu *et al*.,[13] in which cartoon water shapes are placed on templates and animated along a path.

To compare the existing methods for cartoon water animation, we show the visual component, motion style, interaction handling and appearance of particle, fluid simulation and template-based approaches in Table 1.

In this paper, we describe a method for modeling the interaction between objects and the water for cartoon animation. Our method involves the definition of flow path grid in the presence of obstacles, modeling of water forms, and combinations of different forms in space and time, as detailed in the following sections.

Water Path Grid

In cartoon animation, water features usually move along a path that varies according to the scene. In our system we first require users to specify a few nodes to indicate the location of the water path boundary, interpolate these to get the two path boundary curves, and then we interpolate the two boundary lines to get a set of intermediate path lines U_i $(i = 1, \ldots, n)$ to indicate the water flow field that guides the placement and orientation of the movement for water forms. Here n is the number of path lines, we set it to 150 as a default, which is adequate for our modeling task on a window of 800×600 pixels. Next we connect corresponding points between these path lines to get another set of lines V_j $(j = 1, \ldots, m)$ where m is the number of points contained in each path line U_i. With U_i and V_j a grid can be drawn

DOI: 10.1002/cav

Methods	Visual components	Motion style	Interaction handling	Appearance
Particle based	Particles	Realistic or stylized	Yes (because particle is small and external force applied directly)	Particle usage is obvious and dominates
Fluid simulation based	None (invisible cells are used as the unit for fluid simulation)	Realistic (physically based)	Yes (because cell is small and easy to include external forces)	Fluid surface rendered with lines and flat colors
Template based	Stylized water shapes with structures	Stylized (by animating templates which include low frequencies of motion)	No (because templates are not associated with a physical model)	Same liveliness and energy of hand-drawn cartoon water

Table 1. Existing methods for cartoon water animation

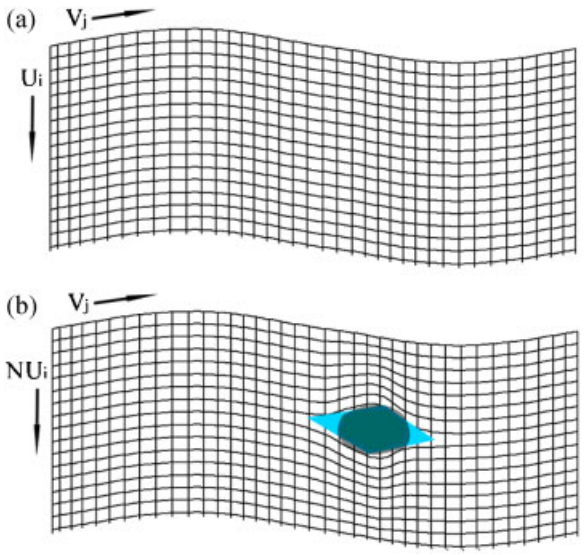


Figure 1. Water path grid: (a) water path lines composed of U_i and V_j; (b) new path lines composed of NU_i *and* V_j.

over the water path (as shown in Figure 1(a)), which now forms as a reference grid over the water path.

When objects such as stones and boat-hulls are placed in the water path, we leave the boundary lines unchanged, and let the intermediate path lines bend around the sides of the object in question. In order to get a smooth transition in U_i from the region where no obstacles are present to the region where an obstacle is present, we use a rhombus to cover the object. The four vertices of the rhombus are then taken as constraints which effect alterations to the path lines U_i, resulting in a spindle shape around the object in the water path (Figure 1(b)). Next we describe how to determine the four vertices of the rhombus.

Our UI allows users to drag object images into the water path and specify a few points on the object image,

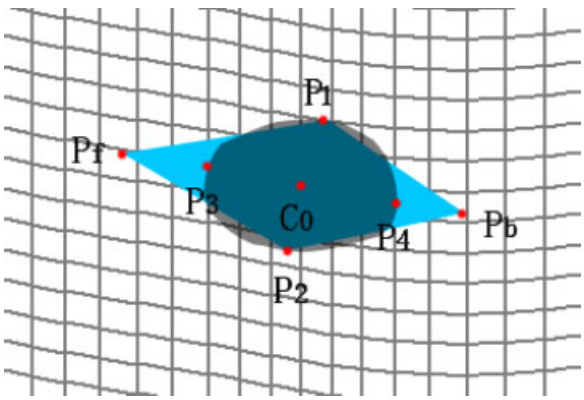


Figure 2. Determination of four vertices of the rhombus.

these points are interpolated by the spline to get a closed curve C_k $(k = 1, \ldots, cn)$ that approximates the cross-section boundary of the object on the water surface, (here cn is the number of points in C_k). We calculate the weighted center C_0 of C_k and search for the grid cells that C_0 falls into. Once the grid cell is found, we take it as a reference to determine the four vertices of the rhombus, as detailed next.

We first determine the two extreme points P_1 and P_2 in C_k facing the two boundary lines by searching those grid cells into which points of C_k fall and select those cells whose i indices have maximum and minimum values. P_1 and P_2 are served as two vertices in the rhombus as shown in Figure 2. The other two extreme points P_3 and P_4 in C_k along the flow direction are determined by searching those grid cells into which points of C_k fall and select those cells whose j indices have maximum and minimum values. We then "push" P_3 and P_4 outward with reference to C_0 by a distance of $f\|P_3, P_4\|$, respectively to get two other vertices P_f and P_b of the rhombus (Figure 2), where f controls the how much the rhombus is "stretched."

DOI: 10.1002/cav

Figure 3. Some hand-drawn water forms.

We interpolate P_f, P_1, P_b and P_f, P_2, P_b with splines to get two curves that draw a spindle shape around the object and, through experiment, we found that $f = 1.5$ will usually produce a good-looking spindle shape for a smooth transition. The two curves drawing the spindle shape are then taken as the new internal "boundaries" and the new intermediate path lines (i.e., NU_i) are obtained by interpolating corresponding parts on the original two boundary lines and the new internal boundaries (Figure 1(b)).

In our system, multiple objects are allowed in the water flow. An example of two stones in a stream is shown in Figure 8 (right) in Subsection "Running Stream With Still Obstacle Objects."

Water Forms

In cartoon animation, water movements are represented with a variety of stylized forms with different structures and shapes, and a few hand-drawn cartoon water forms are shown Figure 3. Based on the observation of hand-drawn cartoon water forms, we classify them into three types: (1) "main waves" depicting the peaks of the waves in the current, (2) scattered forms depicting water textures between the main waves, and (3) forms interacting with objects. We proceed to describe how to model them in the next three sub-sections.

Main Waves

Main waves are characterized by oscillating curves, small ballistic arcs depicting the front profile of the waves (heading the orientation of the movement), and large ballistic arcs depicting the back profile of the waves. A few holes may appear on the main wave forms. The key for successful profile synthesis here is to choose suitable models to define their shapes.

We can represent the oscillating curve by a modular of nodes determined with a simple stochastic model:

$$\mathrm{OCUR}_i = c + \mathrm{rand}(v) \quad (i = 1, \ldots, sn) \tag{1}$$

where OCUR_i is the ith sample point of the wave profile defined on a uniform world space, sn the number of the sample points, c a constant which is the mean magnitude of OCUR_i and $\mathrm{rand}(v)$ is a random variable distributed about zero with a standard deviation of v. Although our model is simple in structure, a number of variants of these curves can be generated by different parameters for c and v as well as the sampling interval. We interpolate these sample points with the spline to get smoothly oscillating curves of the forms shown in Figure 4 (left). During animation, we vary c, v, and the sample interval by small random amounts to simulate the slight imprecision of a hand-crafted look.

We can model the small ballistic arcs straightforwardly using Equation (1). First, we pick up two neighboring points as defined by Equation (1), say OCUR_i and OCUR_{i+1}, and then add a new point P_{in} in between them. We let the magnitude of P_{in} vary randomly in the range $(0.3 - 0.5)\min(\mathrm{SCUR}_i, \mathrm{SCUR}_{i+1})$ and then interpolate OCUR_i, P_{in} and OCUR_{i+1} with the spline to obtain a small ballistic arc SBARC_i, as indicated by the curve above the horizontal axis in Figure 4 (right).

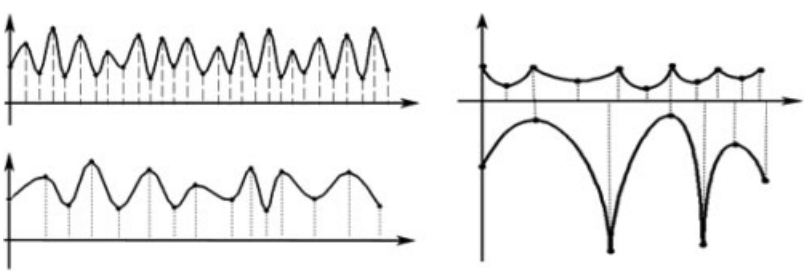


Figure 4. Oscillating curves and ballistic arcs.

DOI: 10.1002/cav

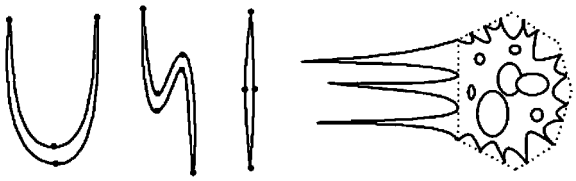


Figure 5. Individual forms and splashing spray.

The large ballistic arcs depicting the back profile of the waves can be modeled in a similar manner by increasing the magnitude of the sample points as well as the interval of ballistic arcs, as shown by the curve below the horizontal axis in Figure 4 (right). We note that the orientation of the ballistic arc is opposite for the front and back profile in the main waves.

Scattered Components (Between Main Waves)

Scattered components depicting ripples between main waves are stylized with U shapes, S shapes, and a linear form with a bit of depth terminated at a shallow angle at each end. These are constructed with a few control points determined by templates like those in Figure 5 (left three). While straightforward to construct their presence and behavior gives life to parts of the water surface whose constant movement is not otherwise represented.

Forms Interacting With Objects

Interactions between objects and the water may produce a variety of water forms under different conditions such as speed of the water flow and size of the object as well as wind conditions. Finding comprehensive models for all these forms requires much research work to ensure the necessary characteristics of stability in animation and stylistic suitability. In our current system we model two types of forms interacting with objects: (1) waves splashing over the object, (2) water forms flowing around a moving object, including bow-waves. We detail how these are modeled next.

Splashing Over a Static Object. In the presence of obstacles to the flow of water, (main) waves are deflected by the objects and create sprays which travels forward from the impact and falls back on to the water surface producing further effects. Such splashing exhibits quite different shapes from those previously mentioned due to the speed of water flow as well as obstacle size. In our current implementation, we model the spray (or spume) covering the obstacle object due to the collision between the water waves and the object.

The front profile of the splash is modeled with SBARC along a convex reference curve constructed by interpolating a number of points sampled on an ellipse with different intervals. Some ellipse-like shapes are then subtracted from the splash profile with varying size and orientation to make holes in the splash spray. A few large ballistic arcs are finally added as a "tail" to depict the back profile of the splash (Figure 5 (right)).

It should be pointed out that this back profile with its large ballistic arcs is not caused by the current action of the splash but is constituted by "left overs" from earlier activity (wave breaking etc). We add them because the path lines NU_i vary their shapes around the objects in the water flow; in other words the area filled by the object expels path lines. As a result, the main wave forms split into two parts which flow along the two sides of the object after encountering it. The splash is augmented with the "tail" of large ballistic arcs (containing bubbles drawn down by the spume) so that the back profile of the splash looks like a stylized form of the bubble paths seen in nature and is then seen to interact with the passing object.

Water Forms Surrounding Moving Objects. In addition to the splash caused by water waves striking against a static object, water flow may also create time-varying forms along the border between the object and the water surface. In this section, we describe how to model these water forms by taking objects such as a boat and an animal moving in the water as examples.

Since the moving object is determined by the scene, we require the user first to drag the object (a boat in Figure 6 for example) into the water path, and then specify a few control points along the visible edge between the object and the water surface (yellow dots in Figure 6). These points are interpolated by the spline to get a reference curve. If the moving object interacts heavily with the

Figure 6. Water forms surrounding a moving boat.

Comp. Anim. Virtual Worlds 2008; **19**: 375–385
DOI: 10.1002/cav

water in front of the object, this will cause a substantial bow-wave stylized with sharp points thrown forward. Meanwhile water forms around the side of the object will be less dynamic because the force applied is along or near to the tangent of the body shape and this will be relatively small. In setting up this situation interactively we divide the reference curve into two segments by clicking a yellow point (which is highlighted in green in Figure 6): SEGHEAD (in red in Figure 6) on which we put a heading profile with a few sharp shapes in the oscillating curves OCUR, SEGSIDE (in pink in Figure 6) on which we put SBARC for the upper profile and a simple, less dynamic curve for the lower profile on the water surface. In order to represent the turbulence of the water along the side of the object, we generate another reference curve below SEGSIDE based on a simple relation to SEGSIDE, and put a new water form on it which is more characteristic of the behavior of water along the side of a boat.

In the case of a fast-moving boat some additional longer forms are added to represent the turbulence behind the boat. These are easy to model with the methods we have used so far. In addition to the water forms surrounding the visible edge between the object and the water surface, a moving object also creates ripples ahead of the object. We model them with two sets of SBARC on the reference curves, representing the outer profile facing the direction of motion and the inner profile facing the object.

Combinations of Different Water Forms in Space and Time

In cartoon animation, all the above-mentioned forms are arranged according to rules covering appearance and behavior. These ultimately determine the look and stylization of the animation. In this section, we proceed to describe the combination of different water forms in two cases: a running stream with still obstacles in it and the water surface local to moving objects. We start with the first case.

Combination of Water Forms in Flowing Water

In the case of a stream containing stationary obstacles, we need to combine main wave forms, scattered water texture shapes between the main waves, and splashing caused by deflection of the water body by spatial and temporal structures, as detailed next.

Main Wave Forms. In a flow of running water we will see a number of main wave forms moving at a certain distance from each other along the given path. We suppose there are mwn main waves in the stream and we derive the distance wd in terms of a grid between two adjacent main waves by dividing m (the number of V_j in the path grid) with mwn, and then select V_j every wd grid point in the path to provide the base lines needed to guide the placement of the main wave forms. Linear base lines are, however, too uniform and regular and the resultant main wave forms attached to them look too mechanical. We therefore use the sum of two sinusoidal components defined on each base line to obtain curved base lines consistent with a particular stylization of nature:

$$\begin{aligned} \mathrm{BLINE}_i = \mathrm{BASE}_i &+ am_1 \times \sin(w_1 + a_1) \\ &+ am_2 \times \sin(w_2 + a_2); \end{aligned} \tag{2}$$

There are three components in this equation. The first component BASE_i refers to the smooth base lines taken directly from the path grid. The second component is a sine-wave function with amplitude am_1, a low frequency w_1 determining the main shape of the curved base line with an added random phase a_1 to avoid the repetition of the curved reference lines which the eye would otherwise pick out. The third sine component has a small amplitude am_2, a higher frequency w_2 and a random phase a_2, describing local details of the curved base line. Our interface allows the user to tune the parameters in Equation (2) to get the desired dynamic curve base lines, as shown by dashed lines in Figure 7.

In hand-drawn main waves the wave top profile is composed of oscillating curves and small ballistic arcs in different places in the profile. We simulate this by combining OCUR, SBARC, and large ballistic arcs as follows. We first divide curved base lines into a small number of segments, and then attach OCUR or SBARC randomly to them as front profiles (the pink and red

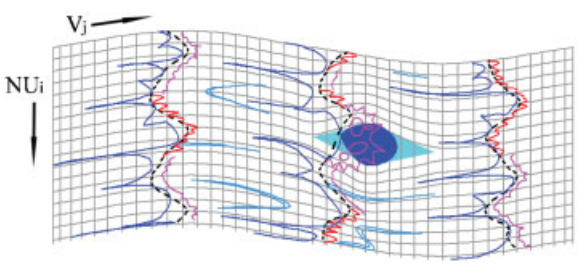


Figure 7. Combinations of different types of forms in the running stream.

DOI: 10.1002/cav

curves in Figure 7), and the larger ballistic arcs as back profiles (the blue curves in Figure 7).

The procedure to generate the main wave forms can be expressed as follows:

```
For each frame of animation do
  {Generate a number of curved base lines BLINE_i in the path grid;
  Attach the front and back profile of the main wave to each base line BLINE_i}
```

Scattered Water Texture Shapes Between the Main Waves. In addition to the main wave forms that dominate movement in the given path, scattered ripples in the U, S, and linear forms may be introduced at random between the main waves and, once seen, their lifetime is as follows. Each shape first grows to its maximum size, then becomes progressively smaller and finally disappears during the following few frames. This behavior is very much like what happens in particle systems where a particle's life is tracked from (simulated) birth to death.[14] In our implementation, we adopt five frames as the life interval for each shape and use an empirical factor $sf = 0.3 + 0.7 \times \sin(\pi \times i/5)$ $(i = 1, \ldots, 5)$ to scale each shape.

In summary, our mechanism for arranging scattered water texture shapes between main waves is given below:

```
//Initialization
For each base line do
  {Determine the number of scattered components stochastically;
  For each component do
    {Determine the Position of the current component relative to the base line stochastically;
    Determine the Type stochastically and the Lifespan for the current component;}
  }
//Begin procedure
For each frame of animation do
  For each base line do
    For each component do
      {If the lifespan of the current component is over then
        {Determine the Position, Type, and Lifespan for a new component;}
        Select a component with corresponding Type;
        Check the life index of the current component and use it to calculate the scaling factor sf;
        Scale the current component with sf;
        Place the current component to its corresponding position.
      }
```

In Figure 7, we use light blue curves to show a resultant arrangement of scattered components between the main waves.

Splash Caused by Deflection From an Obstacle. When we use a constructed splash form (Figure 5 (right)), it is critical to make the splash appear at the right time because it is caused by the deflection of water waves by the object. In our system we detect the point of initial deflection by calculating the distance between the base lines of the main wave forms and the object's previously calculated center C_0 and the splash is added when the main wave forms move close enough to the object for the distance to become smaller than a predefined threshold, as shown by the form in pink close to the obstacle object in Figure 7.

Overall Procedure. The overall procedure for combining different water forms in the path grid is expressed as follows:

```
For each frame of animation do
  {Generate main waved forms;
  Arrange scattered water texture shapes between the main wave forms;
  Detect the distance between the obstacle objects and the base lines;
  If the detected distance is small than the predefined threshold then
    Add the splash spray.
  }
```


DOI: 10.1002/cav

Local Water Surface With Moving Objects

In cartoon animation it is often required to animate objects moving on a local part of the water surface in medium long or medium shot. In this case three types of water forms are used: scattered ripples on the water surface, forms surrounding the object, and ripples pushed out in front of the object. Dynamically, these three types of forms have different behaviors. Scattered forms on the water surface may appear randomly on the water surface to give us a shimmering effect, or move along in a fixed orientation to simulate the camera movement, while forms surrounding the object and ripples ahead of the object move along with the moving object. We use a layered approach to deal with above three types of form in space and time.

In the first layer, a number of ripples are placed over horizontally parallel construction lines. On each construction line, positions of the ripples are controlled by a Boolean function which makes the ripple appear where the function is 1 and disappear where the function is 0. The duration of the positive value in the Boolean function is varied stochastically. An advantage of using a separate layer for the scattered ripples is that we can control them independently of the object movement, say, simulating the camera movement by moving the layer in the appropriate direction during the animation.

Ripples ahead of the moving object not only move with the moving object, but also radiate outward during the movement. The construction lines for ripples are curved and specified by users according to the shape of the moving object, positions of the ripples on each curve are also controlled by use of Boolean functions defined on the scaffolding of the reference curves. The duration of the positive value in the Boolean function varies stochastically, with its mean value decreasing as the ripple moves outward and, inversely at the same time, the duration of 0 in the Boolean function varies stochastically with its mean value increasing.

Since forms surrounding the object, and ripples ahead of the object, move along with the moving object, they may overlap with scattered ripples. Such cases need to be avoided in cartoon animation. In order to avoid such overlaps, we use a polygon to approximate the contour of the area inside the object contour summed together with the area covered by the ripples ahead of the moving object. During animation, our system checks whether the ripples on the first layer overlap with this polygon mask and only draws those ripples outside the mask.

Results

In this section, we present some results of our approach to modeling the interaction between objects and the cartoon water.

Running Stream With Still Obstacle Objects

In Figure 8 (left), we show a strip of six frames sampled from a video of a running stream containing a stone. This shows the effects of the main waves running along a given path as well as the splash caused by the interaction between the main water forms and the stone. To stage the animation properly, a background picture is added in the scene. From these six frames, we can see that water forms deform naturally around the obstacle object. Our water path changing theme (Section "Water Path Grid") allows the object to change in shape and size and,

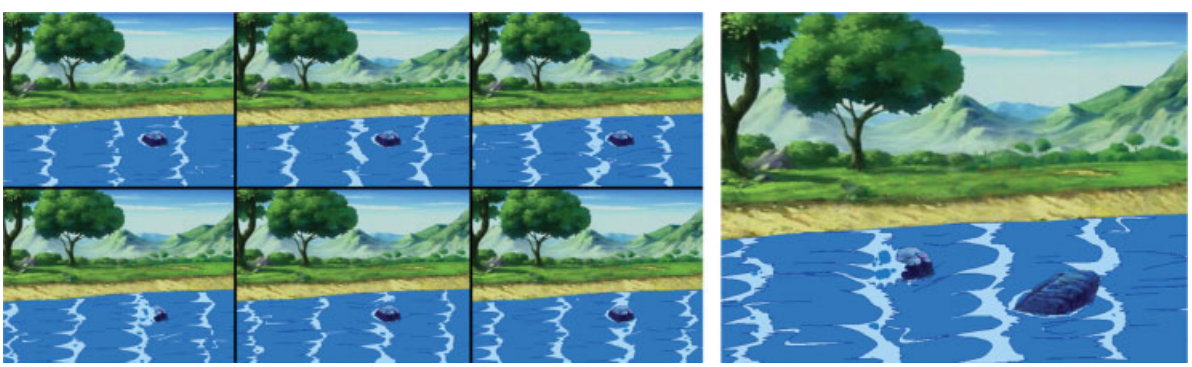


Figure 8. A strip of six frames sampled from a video of a running stream containing a stone (left); a stream running by two stones (right).

Comp. Anim. Virtual Worlds 2008; **19**: 375–385
DOI: 10.1002/cav

more significantly, multiple obstacles are acceptable in the water path. An example of two stones in the running stream is shown in Figure 8 (right).

In the accompanying video we also show that the user is able to tune parameters easily through our UI to alter the speed of the movement, water wave spacing etc. The system gives instantaneous visual feedback and animates water effects in real-time.

Local Water Surface With Moving Objects

In this section, we show two examples of objects moving on a local water surface, giving an impression of a medium long shot for animation.

The first example is a boat moving horizontally to the left side of the screen (Figure 9 (top)). Some ripples are scattered on the water surface in a consistent manner.

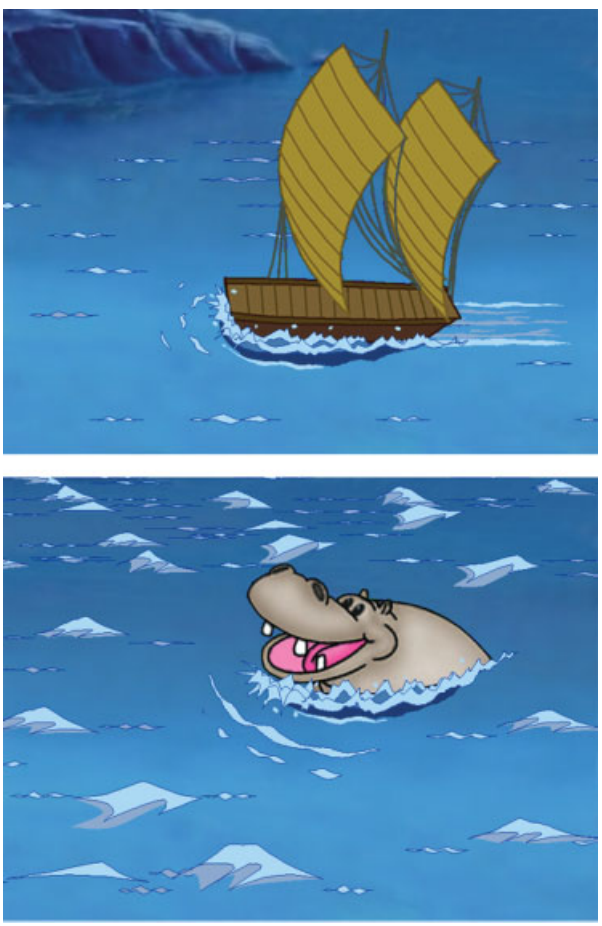


Figure 9. A moving boat (top); a moving hippopotamus (bottom).

Figure 10. Animating boundary lines (top three); "Casa" represented with cartoon water (bottom).

In the second example we change the object into a hippopotamus which moves in a right–left direction toward the viewer (Figure 9 (bottom)). Scatted ripples on the water surface are moved rightward at a constant speed to simulate the camera "movement" in the animation, as shown in the accompanying video.

In addition to animating turbulence between objects and cartoon water, the path driven theme in our method allows users to use boundary lines to make unanticipated effects. Three shots from an animation are shown in Figure 10 (top), which show cartoon water initially animated on a sphere which then transforms into a stair. In Figure 10 (bottom) our last example shows cartoon water being animated to form the words "Casa." A shadow has been added to give a 3D sense to the scene.

Conclusions and Future Work

In this paper, we have presented the results of our research on cartoon water effects, with a focus on the interaction between flowing water and obstacles. There are several particular features with our method:

(1) Easy setting with little user intervention. In our system, users are required to make simple interactions to provide initial settings on the model, the detailed water forms associated with settings are generated automatically by the model.

Comp. Anim. Virtual Worlds 2008; **19**: 375–385
DOI: 10.1002/cav

(2) Complex water forms. Water forms modeled in this paper are more complex than those in previous works, and here we are able to animate more violent behavior in cartoon water.
(3) Adaptiveness to changes in obstacles. Our mechanism for path line definition allows a dual-mode water effects animation with and without obstacles. In the presence of obstacles, changes in object shape, size, and even the number of still objects can be dealt with adaptively. We can also simulate camera movement as well as the orientation of the movement to track moving objects straightforwardly.

In our future research, we intend to model more cartoon water forms including ocean waves under different wind conditions, sea surfaces as they approach the littoral, etc.

ACKNOWLEDGEMENTS

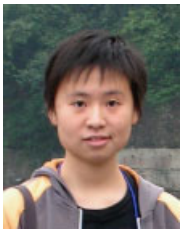
We thank anonymous reviewers for their suggestions and comments. This work is supported by National Key Basic Research and Development Program of China (973 program) (No. 2002-CB-312101), the National Natural Science Foundation of China (No. 60673007), and National 863 High Tech. Research and Development program of China (No. 2006AA01Z312).

References

1. Whittaker H, Halas J. *Timing for Animation*. Focal Press Limited: London, 1981.
2. Bregler C, Loeb L, Chuang E, Deshpande H. Turning to the masters: Motion capturing cartoons. In *Proceedings of SIGGRAPH*, 2002; 399–407.
3. Iglesias A. Computer graphics for water modeling and rendering: a survey. *Computer Graphics Forum* 2004; **20**: 1355–1374.
4. Adabala L, Manohar S. Techniques for realistic visualization of fluids: a survey. *Computer Graphics Forum* 2002; **21**: 65–81.
5. Yu J-H, Patterson J. A fire model for 2d computer animation. In *Proceedings of Computer Animation and Simulation*, 1996; 49–60.
6. Chenney S, Selle A, Mohr A. Cartoon rendering of smoke animations. In *Proceedings of the 3rd International Symposium on Non-photorealistic Animation and Rendering*, 2004; 57–60.
7. Di Fiore F, Claes J, Van Reeth F. A framework for user control on stylised animation of gaseous phenomena. In *Proceedings of Computer Animation and Social Agents*, 2004; 171–178.
8. Hartnett C, McGuire M, Fein A. Real-time cartoon rendering of smoke. *Poster 0067, SIGGRAPH*, 2004; 171–178.
9. Fein A, McGuire M. Real-time rendering of cartoon smoke and clouds. In *Proceedings of the 4th International Symposium on Non-photorealistic Animation and Rendering*, 2006; 21–26.
10. Xu D, He H. Real-time rendering of cartoon smoke and clouds. *Computer Animation and Virtual Worlds* 2005; **16**: 441–449.
11. Thornton JD. Directable simulation of stylized water splash effects in 3d space. Sketches & Applications, *SIGGRAPH*, 2006.
12. Eden AM, Bargteil AW, Goktekin TG, Eisinger SB. O'Brien JF. A method for cartoon-style rendering of liquid animations. In *Proceedings of Graphics Interface*, 2007; 51–55.
13. Yu J-H, Jiang X-N, Yao C, Chen H-Y. Real-time cartoon water animation. *Computer Animation and Virtual Worlds* 2007; **18**: 405–414.
14. Reeves WT. Particle systems—a technique for modeling a class of fuzzy objects. In *Proceedings of SIGGRAPH*, 1983; 359–376.

Authors' biographies:

Dr Jinhui Yu is a professor of computer science at Zhejiang University, China. He received his PhD in computer graphics from the University of Glasgow in 1999. His research interests include stylized computer animation and computer graphics art.



Jing Liao is currently a post graduate student in the state key lab of CAD&CG, Zhejiang University, China. Her research interests include computer animation and non-photorealistic rendering.

John Patterson is a Research Fellow in Computing Science at Glasgow University, specialization in drawn

DOI: 10.1002/cav

animation and the extension of cartoon technology to image manipulation, and film effects technology generally. His academic collaborations include the MTRC at Bath University, Rainbow group at Cambridge University the EDM, University of Hasselt (Belgium), and the State Key lab. of CAD & CG Hangzhou (P.R. China). His industrial collaborations include work with Cambridge Animation Systems, NFTS CREATEC, Anthropics, Androme (Belgium), and AmaK Studios (France). He was project coordinator 1994–1996 for the ANIMAX project in the Computer CARTOON program and for the IST project CUSTODIEV (2002–2005).

Comp. Anim. Virtual Worlds 2008; **19**: 375–385
DOI: 10.1002/cav