

Christopher M. Smemoe

C S 555

Paper 2

Flow Trace Animation

Introduction

It is difficult, if not impossible, to model some phenomena with spheres, polygons, or other objects and to render them using standard rendering techniques. Fire, water, smoke, and gases stand among these phenomena that are difficult to model. However, in recent years, research has been performed so these “fluid-type” phenomena can be effectively visualized. In 1983, William T. Reeves presented a method of modeling this class of fluid objects. Reeves referred to these objects as “fuzzy” objects—fuzzy objects do not have “smooth, well-defined, and shiny surfaces; instead their surfaces are irregular, complex, and ill-defined” (Reeves, 1983).

Reeves presented a particle-based method of animating fuzzy objects that differed from traditional rendering and animation in three ways. First, clouds of particles represented objects in Reeve’s method. These particles were used instead of using polygons or curved surfaces to define the object’s location. Second, each particle had a lifetime during which it grew in intensity while moving along its path. Some particles were born at each frame of the animation while others died. Finally, random processes were used to determine the shape and appearance of an object in Reeves’ method of fuzzy object animation.

Five steps are involved at each animation frame in Reeves’ method. First, new particles are randomly generated into the system. Second, each new particle is assigned individual attributes such as a position, velocity, size, color, transparency, shape, and a lifetime. Each of the particle attributes can be held constant or can be varied throughout the lifetime of a particle. Third, any particles that have existed past their prescribed lifetime are “killed off”. Fourth, any particles not killed off are transformed according to their individual static or dynamic attributes. Finally, an image of all the existing particles is rendered in a frame buffer.

Around 1990, Reeves’ approach was extended to fluid flow visualization by van Wijk (see van Wijk, 1990). Van Wijk presented a simple particle-based method of animating two-dimensional flow. Then, in 1993, Jones and Saito improved upon van Wijk’s method of two-dimensional flow visualization.

This paper will focus on Jones and Saito’s two-dimensional flow vector field animation algorithm (Jones and Saito, 1993). Jones and Saito call this flow animation *flow trace animation*.

Flow Trace Animation

The purpose of flow trace animation is to visualize flow velocity fields given a two-dimensional mesh or grid of nodes with x and y velocity values at each node. In flow trace animation, a set of x-y points in the model is randomly chosen. Each of these x-y points is then tracked through a given number of animation frames. The particles are moved based on the velocity field. In regions of high velocity, long flow traces are generated, while shorter flow traces are generated in regions of low velocity. Jones and Saito liken the flow traces generated using their algorithm to a meteor shower. An image resulting from a flow trace animation is shown in Figure 1.

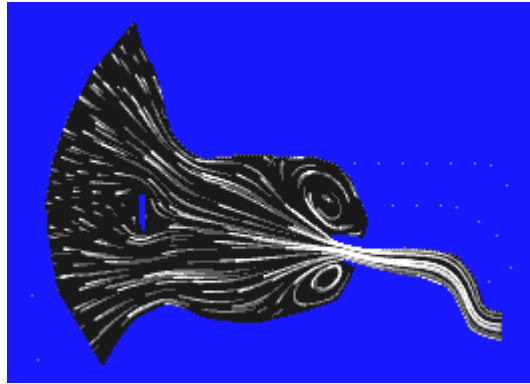


Figure 1: A flow trace animation of a river entering into a bay. From the Surfacewater Modeling System (SMS) home page, 1995.

Application of Particle Systems to Flow Trace Animation

Flow trace animation combines ideas from William T. Reeves' original paper on particle systems, J.J. van Wijk's work in flow visualization using particle systems, and Korein and Badler's (1983) work in motion blur techniques in animation. In addition, the flow paths of each particle are drawn as Bezier curves to make the flow paths appear more natural.

As was mentioned in the introduction, five steps are involved in modeling and rendering particle systems. For each frame, new particles are first randomly generated into the system. The number of new particles in a frame is determined according to Equation 1 (Reeves, 1983):

$$NParts_f = MeanParts_f + Rand \times VarParts_f$$

Equation 1: Determining the number of particles in a frame of the flow trace animation.

Where $MeanParts_f$ is the average number of particles in a frame, $Rand$ is a function that returns a number between -1.0 and 1.0 , and $VarParts_f$ is the variance of the average number of particles.

Second, each new particle is assigned a position, velocity (both magnitude and direction), intensity, and a lifetime. In flow trace animation, the velocity of a particle is interpolated from the nodal velocity values. Third, any particles that have existed past their prescribed lifetime are "killed off". The prescribed lifetime is a number of frames determined from the velocity of the particle at its initial location and decremented with each passing frame.

Any particles not killed off are translated along a Bezier curve according to their individual static or dynamic attributes. The motion of all the existing particles is then anti-aliased and rendered in a frame buffer.

An Algorithm for Computing a Flow Trace Animation

Jones and Saito use basic ideas from particle system theory to compute a flow trace animation that displays the results of a transient surface water model. In their computations, each trace starts out dimmed, becomes brighter until it reaches its maximum intensity, then becomes dimmer before being killed. This makes the animation resemble a meteor shower (Jones and Saito, 1993).

Step 1: Rasterize the Animation Area

First, rasterize the area of interest and perform the computations on a pixel-by-pixel basis. Interpolate and store the x and y flow velocities at each pixel.

Step 2: Determine Initial Locations of Flow Traces

Use a random number generator to compute a set of locations on the mesh where new flow traces can be initiated. The number of new flow traces is determined according to Equation 1. Determine the frame lifetime of each flow trace from the velocity magnitude at the initial location of the flow trace.

Step 3: Anti-Alias and Render the Pixels through Which Each Particle Moves

For each frame, use the pixel velocities to compute the direction of the increment added to the path by the flow trace. Use the magnitude of the velocity to determine the length of the increment. A motion blur technique is used to create a “blurred” animation effect (Korein and Badler 1983). Use van Wijk’s method to determine the intensity of each pixel (1990). In his method, van Wijk determines the intensity of each pixel according to the following equations:

$$u = (t - t_0) \bmod T$$

Equation 2: Determination of u , the current time of a particle with respect to T , the time between two frames. t is the current time of the particle in the animation, while t_0 is the initial time of the particle in the animation.

$$I(t) = \begin{cases} 2I_{\max} u/T & \left\{ u \leq T/2 \right. \\ \left. 2I_{\max} (1 - u/T) \right\} & \left. u > T/2 \right. \end{cases}$$

Equation 3: Determination of $I(t)$, the pixel intensity at time t . I_{\max} is the maximum pixel intensity, and u and T are defined above.

Determine the path of each flow trace using a Bezier curve. A Bezier curve produces a continuous curve along the direction of the flow trace. Anti-alias the pixels along the Bezier curve according to techniques described in Foley et al. (1990).

Step 4: Kill or Extend Existing Flow Traces and Initiate New Flow Traces

For the next frame, extend or kill off existing flow traces and initiate new flow traces using the random number generator. If a flow trace’s lifetime is zero, the flow trace is killed off. Each flow trace’s lifetime is decremented by one frame. According to step 2, new flow traces are then generated and their lifetime is set. All the current flow traces are then rendered according to step 3. This process is continued until a user-specified time is reached.

Conclusions and Further Research

Flow trace animation is a highly effective method of visualizing two-dimensional surface water flow. The results of a flow trace animation resemble traces left by meteors in a meteor shower. Areas of high velocity leave long flow traces that gain intensity until a maximum intensity is reached, after which the intensity dies out. And areas of low velocity leave short flow traces.

In flow trace animation, Reeves' particle-based rendering methods are used in combination with Bezier curves, anti-aliasing techniques, and stochastic motion-blur and particle spacing methods to produce good animations of fluid flow.

Future research could include extending the two-dimensional flow trace algorithm to three dimensions. In addition, flow trace animation could be used to model other types of fuzzy objects, including fire, smoke, gases, and other fluids. Equations could be generated that could effectively be used in flow trace animation to model several types of phenomena.

For example, if the velocity of a fluid is not known at locations in a model but other parameters are known, perhaps the velocity could be computed at each location in the pixelized model. One equation for computing velocity along a stream channel is Manning's equation:

$$V = \frac{1.49}{n} R_h^{2/3} S_e^{1/2}$$

Equation 4: Manning's equation—an equation that can be used to compute the velocity at each location in the pixelized model. V is the velocity, R_h is the hydraulic radius at the point, S_e is the slope of the channel at the point, and n is Manning's n-coefficient.

After the velocity at each pixel in the model is computed, these velocities could be used to compute a flow trace animation.

References

- Alan Watt and Mark Watt, “Advanced Animation and Rendering Techniques”, ACM Press, New York, New York, pp. 415-417:*Particle set animation*, 1992.
- James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes, “Computer Graphics: Principles and Practice—Second Edition”, Addison-Wesley Publishing Company, Reading, Massachusetts, pp. 1031-1034:*Particle Systems*, also pp. 488-491:*Bezier Curves*, 1990.
- Jarke J. van Wijk, “A Raster Graphics Approach to Flow Visualization”, *Proc. Eurographics*, pp. 251-259, September 1990.
- J. Korein and N. Badler, “Temporal Anti-Aliasing in Computer Generated Animation”, *Proc. Siggraph*, pp. 377-388, 1983.
- Norman L. Jones and Takafumi Saito, “Flow Animation Techniques for Two-Dimensional Hydrodynamic Modeling”, *Advances in Hydro-Science and Engineering*, vol. 1, part B, pp. 2091-2096, June 1993.
- Thierry Delmarcelle and Lambertus Hesselink, Richard S. Gallagher, Editor, “Computer Visualization: Graphics Techniques for Scientific and Engineering Analysis”, Chapter 5:*A Unified Framework for Flow Visualization*, CRC Press, Boca Raton, Florida, pp. 139-142:*Particle traces, streaklines, and streamlines*, 1995.
- William T. Reeves, “Particle Systems—A Technique for Modeling a Class of Fuzzy Objects”, *ACM Transactions on Graphics*, vol. 2, no. 2, pp. 91-108, Apr 1983.
- William T. Reeves and Ricki Blau, “Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle Systems”, *Computer Graphics*, vol. 19, no. 3, pp. 313-322, July 1985.