Interactive Flow Field Modeling for the Design and Control of Fluid Motion in Computer Animation

by

WILLIAM FRANKLIN GATES

B.Sc., The University of California at Davis, 1991

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF COMPUTER SCIENCE

We accept this thesis as conforming to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

January, 1994

Abstract

Ubiquitous and captivating, fluid—gaseous and liquid—motion is often desired in computer animation. For example, rives, lakes, and clouds enhance flights simulations. Realizing realistic fluid behaviour, however, can be difficult and laborious using traditional computer animation methods. Ad hoc kinematic models of fluid motion have been presented to facilitate the animation of fluids, but it is not clear how to extend or integrate these models to describe more general fluid motion. Simple dynamic models have been presented, but it is difficult to control the dynamic simulation to achieve the desired effect. To address this problem, a simple, hydrodynamically-based framework for realistically integrating models of fluid flow for computer animation purposes is presented. This framework is based on the continuity equation for incompressible flow, and allows flow fields to be linearly combined, regardless of whether they are interactively modeled or computed by dynamic simulation. Novel interactive flow field modeling methods are introduced to allow the animator to manipulate spline curves that correspond to streamlines in the flow field. The spline-based flow fields can be computed at interactive speed on standard graphics workstations. Many dynamic simulations produce flow fields that satisfy the continuity equation, and these can be linearly combined with the modeled flow field to define a mean flow field which is sampled at the nodes of a lattice. Turbulence is modeled by advecting stochastic distributions of models of vortex flow with the mean flow, allowing infinite resolution for small-scale complexity. Geometric models are advected using the final resulting flow field. A simple animation system incorporating the interactive flow modeling methods was implemented and shows this approach to be a promising and easily extendable method of realistically animating fluids.

Table of Contents

List of Figures

Acknowledgements

I would like to thank my supervisor, Dr. Alain Fournier, for the freedom to explore, the support to persevere, and the guidance to succeed. The second reader, Dr. David R. Forsey, has my appreciation not just for his reviewing of this thesis, but for being a source of encouragement from the beginning. The student reader, Mr. Christopher G. Healey, deserves acknowledgment and several mochas for his many helpful comments.

A number of friendly colleagues also went out of their way to provide valuable assistance, including Dr. Pierre Poulin who offered a fountain of helpful references, Mr. Christopher A. Romanzin who parted the Red Sea of writing a thesis using the LaTeX document preparation system, and Mr. Kevin W. Arthur, Mr. Andrew D. Naiberg, Mr. Larry Palazzi, and Mr. Torre D. Zuk who helped drown thesis ennui in Canterbury.

Finally, I would like to acknowledge the financial support provided by Computer Science Department of the University of British Columbia through teaching assistantships and by the National Science and Engineering Research Council through research assistantships that made this research possible.

To
my father,
a place called Cascadia,
and
a love of water.

Chapter One

Introduction

Water, water, every where,
And all the boards did shrink;
Water, water, every where
Nor any drop to drink.

—Samuel Taylor Coleridge

Steam wisping off a rain-soaked tree warming in the sun, bathwater whirling down a drain, a waterfall cascading down a cliff—fluid motion is complex, mysterious, and beautiful. For ages, artists have been trying to capture this beauty, and scientists have been trying to model this motion. Recently, computer animators—both artists and scientists—have attempted to realize the motion of fluids. But while computer animation methods have proven to be a powerful tool for visually representing rigid, human-constructed objects and are routinely used in industry, animating fluids has proven more difficult (Wyvill et al., 1986).

1.1 Motivation

Given the sheer ubiquity of fluids (gases and liquids), the computer animation of flowing fluids has almost as many applications as computer animation itself. When the captivating motion of fluids is also considered, the computer animation of fluids needs little justification. Applications include:

- design of fountains, artificial waterfalls and streams,
- adding models of rivers, lakes, and clouds to visualizations of geographic databases,
- enhanced realism in simulation systems such as flight simulators,
- artistic tools,

- creating television and motion picture visual effects, and
- visualization of fluid phenomena in general.

The wide range of applications motivate the development of animation tools to assist the animator in creating animations of fluid flow. These applications—besides justifying the problem—help define the problem, i.e., motivate criteria for such animation tools.

1.2 Defining the Problem

Creating a computer animation can be loosely divided into two steps: *modeling*, building geometric models and designing their movement and *rendering*, the synthesizing of images from these geometric models using an illumination model, a model of how light interacts with the geometry. Many challenges remain in both the modeling and rendering of fluids. This thesis addresses the motion modeling aspect of animating fluids. In general, geometric modeling and motion modeling cannot be decoupled as the geometry may constrain the possible motions (if realistic motion is desired). However, fluids are continuously deformable, and thus lack any geometric constraints on their motion.

Modeling fluid motion is an extremely broad problem. The range of flowing fluid phenomena can be difficult to encompass: flowing fluids include dripping honey, clouds blowing in the wind, tsunamis, rain, fire, flowing beer, and galaxy formation. Yet, a common set of physical principles underlie these phenomena and motivate a unified approach to modeling their motion.

To make the problem addressed by this thesis concrete and tractable without narrowing its scope, criteria for an ideal solution are defined, a framework for this solution is presented, and the aspect of this framework addressed by this thesis is discussed.

1.2.1 Modeling Motion

The classic method of motion modeling in computer animation is *keyframing* where the positions of geometric models are positioned at particular frames (keyframes), say frame one and frame ten, and the positions at frames two through nine are computed by interpolation. Realistically animating complex fluid motion such as a breaking wave this way would be extremely difficult. Another traditional method of modeling motion is to design motion paths, often using cubic B-splines, that specify the path an object follows. Motion paths are often adequate for models of rigid bodies, but become more difficult for deformable models, and impractical for fluids.

An alternative to keyframing and using motion paths is to use functions that describe the shape of the object over time. For example, traveling sine waves can be used to approximate water waves. Computer graphics researchers have presented ad hoc models for particular fluid motions, but these approaches are limited when more general motion is desired as ad hoc models cannot in general be combined in a straightforward fashion.

A more recent approach to modeling motion is dynamic simulation: forces are modeled, and from Newton's second law of motion, $\mathbf{f} = m\mathbf{a}$, the acceleration is computed and integrated to describe the motion of the geometric model. For example, classical mechanics has been used to animate rigid bodies (Hahn, 1988). This approach has been quite successful, since it gives the animator a high-level control over the motion: the forces. The basic challenges of so-called "physically-based modeling" are developing a dynamic model, finding an efficient method of solving it, and developing some method of controlling the motion. Motion control is essential since dynamic simulation usually defines an initial-value problem that is time-evolved, and realizing a desired motion requires guessing the proper initial conditions and specification of external forces.

The problem with a physically-based model for flowing fluid is that while very good models of fluid motion exist, e.g., the Navier-Stokes equations, the numerical solution is generally computationally intensive and often difficult to set up, especially for free-surface flow problems.

Computer graphics researchers have opted for simplified dynamic models. Perhaps owing to the complexity of fluid motion, relatively little work has been done: a general method of animating fluids remains a challenging problem.

1.2.2 Criteria for an Ideal Method

To alleviate this difficulty, criteria are defined for an ideal method that allows different motion models to be integrated. The applications of the computer animation of fluids motivate the following criteria for such a method:

- *realism,* the motion should be realistic at the relevant scale,
- *efficiency,* the method should be fast enough to be useful,
- *controllability,* the motion computed by dynamic simulation should be easily controlled by the animator, and
- *generality,* the method should not be ad hoc.

The realism criterion, dictated by applications such as flight simulation, means the animation method should allow the creation of fluid models that *appear* to behave like real fluids at the relevant scale. In other words, there is no need for the motion to be in agreement with real fluids beyond what can be perceived in the computer animation. Formally defining a metric for realism, however, is a complex psychophysical problem that has not been fully addressed.

The efficiency criterion means the method should be fast enough to allow interactive animation on standard graphics workstations. As rendering algorithms such as raytracing and radiosity cannot run at interactive rates on most graphics workstations, interaction will have to occur in a "preview mode" using a simple rendering method. Of course, the actual speed is highly dependent on the type of machine, network load, swap space, and other variables. A possible heuristic is that on the order of hundreds of samples used to represent the fluid (vertices, control points, particles) should be animated at interactive rates (certainly no slower than one frame per second). Implicit in this criterion is that any numerical solution method also be stable over relatively long time steps (on the order of 1/30 second, a typical time between animation frames): a fast method that requires very short time steps to be stable is not efficient.

The controllability criterion means the animation method should allow the animator to easily create fluid flow where it is desired and control it at will. This criterion addresses an often noted problem with physical simulation for computer animation: the equations describing the physical phenomena are usually time-evolved from initial conditions and thus getting the desired behaviour can require numerous simulation trials.

The generality criterion means the animation method should be flexible, open-ended, and allow different motion models to be realistically integrated. Consider creating a computer animation of a fountain. A liquid jet might emerge upward, continuous at first, then breaking into discrete parts, then into drops and mist, falling back on itself, and then into a pool where circular waves emanate from each drop, interfere with another, refract with changes in depth, and reflect off boundaries. While ad hoc models could cover separate parts of the modeling, e.g., circular wave motion or drops moving under a gravitational force, ad hoc models are often difficult to integrate, and while two models may be physically valid, there may be no guarantee that their combination is also physically valid. Ideally, we would like models that describe as many different phenomena (waves, jets, spray) as possible to minimize the number of tools the animator must become familiar with and use to create any particular animation.

1.2.3 A Framework for Animating Fluids

The criteria just defined pose a large and challenging problem. Rather than introduce another ad hoc model for a specific fluid phenomena, a framework for solving the general problem is proposed and a subset of this framework is addressed by this thesis.

The framework consists of three major components: dynamic simulation, interactive flow field modeling, and turbulence modeling (Figure 1.1). A flow field computed by dynamic simulation

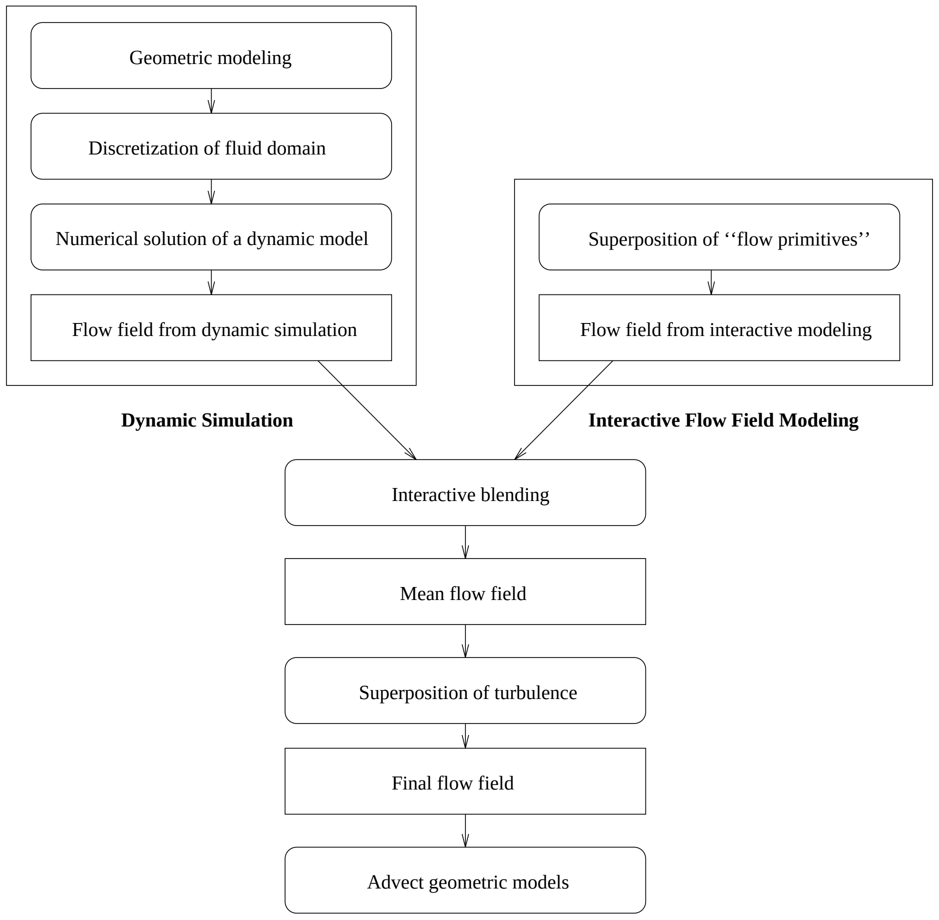
Geometric modeling
Discretization of fluid domain
Numerical solution of a dynamic model
Flow field from dynamic simulation
Superposition of ''flow primitives''
Flow field from interactive modeling
Dynamic Simulation
Interactive Flow Field Modeling
Interactive blending
Mean flow field
Superposition of turbulence
Final flow field
Advect geometric models

Figure 1.1: Framework for an ideal animation method for fluids.

and a flow field interactively model are blended to define a mean flow field. A turbulent flow field is superimposed on the mean flow field.

This thesis presents new methods for interactive flow field modeling using "flow primitives" based on cubic B-splines rather than point primitives and presents a method for using these flow fields to control the flow fields computed by dynamic simulation. A novel turbulent flow modeling method is also introduced.

1.3 Thesis Overview

Chapter two reviews previous work in modeling flowing fluid phenomena. Chapter three briefly discusses basic principles of fluid mechanics to provide a background for subsequent chapters. In chapter four, interactive flow field modeling methods are presented. They are based on the law of conservation of mass as expressed by the continuity equation for incompressible flow. Solutions of this equation are known as “flow primitives” and can be linearly combined. Point primitives (e.g., source, sink, vortex) are reviewed, and novel flow primitives based on cubic B-splines are introduced. Also within this framework, a new method of modeling turbulent flow is introduced as well as a method of controlling the dynamic simulation of fluid flow. In chapter five, a simple animation system incorporating these integrated models is described, and resulting animations are presented. And finally, in chapter six, the advantages and disadvantages of this method are discussed and possible future work is suggested.

Chapter Two

Previous Work

GUARD: 'Allo. 'Oo is it?

ARTHUR: It is King Arthur, and these are my Knights of the Round Table... Go and tell your master that we have been charged by God with a sacred quest. If he will give us food and shelter for the night, he can join us in our quest for the Holy Grail.

GUARD: Well, I'll ask 'im but I don't think 'e'll be very keen... Uh, 'e's already got one, you see?

ARTHUR: What?

LANCELOT: He says they've already got one!

ARTHUR: Are you sure he's got one?

GUARD: Oh yes, it's ver' nice...

—*from* Monty Python's The Holy Grail

Computer graphics researchers have modeled a variety of fluid phenomena. Early work focused on water waves, using simple stochastic models for rough ocean surfaces (Perlin, 1985) and simple kinematic models such as traveling sinusoids for progressive waves (Max, 1981). Later work introduced more sophisticated models for water waves including a stochastic model employing an empirical model of the power spectrum of a rough sea (Mastin et al., 1987), kinematic models describing changes in the shape of waves with varying ocean depth. (Fournier and Reeves, 1986; Peachey, 1986; T'so and Barsky, 1987), and a dynamic model describing wave reflection and net transport of water (Kass and Miller, 1990). More recently, particle systems (Reeves, 1983) where the particles are point masses that do not interact with each other have been used to model ship wakes (Goss, 1990) and waterfalls (Sims, 1990). Interacting particle systems, where the interaction is based on molecular dynamic models, have been used to model highly viscous liquids (Miller and Pearce, 1989; Terzopoulos et al., 1989; Tonnesen, 1991). Despite considerable success with these approaches, animating fluids in general remains a challenging problem.

2.1 Water Waves

Water waves have been modeled by stochastic, kinematic, and dynamic models. Stochastic motion models use random or probabilistic methods to add complexity where deterministic motion is not required. The small scale (relative to the image) of the waves often allows stochastic techniques to be used to bump map a surface model, possibly driven by a kinematic or dynamic model.

Perlin (1985) modeled ocean waves by stochastically perturbing surface normals (bump mapping) according to a superposition of randomly distributed spherical wavefront sources. A random spatial frequency f is assigned to each spherical wavefront source. The amplitude of the wavefronts are $1/f$, and the phase of the sources is modulated by a function of $\sqrt{f}$.

Mastin, Watterberg, and Mareda (1987) used Fourier synthesis with an empirically-based model of the power spectrum of a fully-aroused ocean surface. The algorithm consists of the following steps:

1. Generate a white-noise image.
2. Fast Fourier Transform (FFT) the image.
3. Modulate spatial frequencies based on an empirical model of the power spectrum.
4. Inverse FFT the image.
5. Generate a height field based on image values.

The anisotropic filter attenuates frequencies in one direction and passes frequencies in a perpendicular directions, giving long-crestedness. Motion is simulated by manipulating the FFT phase.

An alternative approach to modeling water waves is to use a model of the changing geometry of the liquid surface. For the following review, a common notation is used: the XY plane models the ocean at rest and the positive Z axis points out of the earth. When height fields are used, $\eta(x, y, t) = z$ is the height of the ocean.

Max (1981) used superimposed traveling sinusoids

$$\eta(x, y, t) = \sum_{i=1}^{n} a_i \sin(p_i x + q_i y - \omega_i t)$$

where (p_i, q_i) is related to the direction of sinusoid i. Linear small-amplitude wave theory gives a (traveling) sine wave as a solution of the linearized equations. The animator specifies the free parameters. Ts'o and Barsky (1987) extended Max's approach to include the effects of wave refraction by means of "wave-tracing", i.e., pre-computing the effect of refraction due to changing depth using Snell's law for a given sinusoid and direction.

Peachey (1986) used a similar model but changed the shape of the wave according to *wave steepness* and ocean depth. The height is given by

$$\eta(x, y, t) = \sum_{i=1}^{n} a_i w_i(\text{fraction}\,[\theta_i(x, y, t)])$$

where w_i is a linear waveform with amplitude a_i. The phase function is

$$\theta_i(x, y, t) = \theta_i(x, y, t_0) - \frac{t - t_0}{T_i}.$$

It depends on the cumulative effects of the depth of the water from the wave origin; these effects are calculated using numerical integration of a function relating the phase to variable depth. The wave profile function,

$$w_i(u),\ \ 0 \leq u < 1,$$

w_i ranges over the interval $[-1, 1]$, with the crest at $w_i(0) = 1$. The wave profile function is linearly blended between a sinusoidal function,

$$w_i(u) = \cos(2\pi u),$$

and a quadratic function,

$$w_i(u) = 8|u - \frac{1}{2}|^2 - 1,$$

based on wave steepness, $S = H/L$. When the depth is small, the parameter u is exponentiated to shift its values toward zero. This has the effect of steepening the front of the wave crest and stretching out the back of the crest (for waves beginning to break). Spray is modeled with particle systems.

Fournier and Reeves (1986) modeled the ocean surface as a parametric surface and modeled wave motion by orbital motion of points on the surface. Unlike a height field representation, the parametric surface representation allows waves to curl over. The equations of the motion of a particle are

$$x = x_0 + r\sin(\phi)$$

$$z = z_0 - r\cos(\phi)$$

where the phase angle is $\phi = \kappa x_0 - \omega t$. To model the effect of wind on top of the crests, the phase angle is modified as

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

where Δz is the height of the wave above the sea level at rest, Δt is displacement in time, and λ is a constant of proportionality. This modification causes a given point to accelerate at the top of the orbit and decelerate at the bottom. Accounting for depth allows for refraction of wavefronts so that waves approaching a beach (with a shallow bottom) take the shape of the shore. The model relates wavelength to the depth h by:

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

The effects of depth on the orbit of a point depend on the effects of depth of all other points before that point: the phase delay is cumulative. It is modeled by:

$$\phi = \omega t + \sum_{0}^{x_0} \frac{\kappa_\infty}{\sqrt{\tanh(\kappa_\infty h(x))}} \Delta t$$

Breaking waves are modeled using

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

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

where

$$S_x = \frac{1}{1 - \mathrm{e}^{-\kappa_x h}}$$

$$S_z = S_x \left(1 - \mathrm{e}^{-\kappa_z h}\right)$$

causing the orbits to become elongated in the direction of the wave motion. Stochastic bump mapping is used to model small-scale waves. Foam and spray are modeled by non-interacting point masses, with initial position and velocity based on the wave shape and celerity (speed) according to a physically-based model. Both are time-evolved according to Newton's second law with a gravitational force.

Deterministic motion can also be described dynamically, i.e., describing the forces and computing the change in the shape of the liquid surface using Newton's second law of motion. This means only the initial conditions (mass, position, and velocity) and the forces need to be specified.

Kass and Miller (1990) model fluid flow by simplifying hydrodynamic equations to the two-dimensional wave equation and then numerically solving it to animate a height field. This model describes wave refraction, reflection, and net transport of the liquid. The model also describes changing topology subject to the limits of a height field representation.

To make the model amenable to a rapid and stable numerical solution, a number of assumptions are made:

- The liquid surface can be represented by a height field.
- The vertical component of the liquid velocity can be ignored.

- The horizontal component of the liquid velocity is small and approximately constant.
- The depth varies slowly.

With these assumptions, hydrodynamic equations of motion and continuity can be approximated by the two-dimensional wave equation

$$\frac{\partial^2 h}{\partial t^2} = g d \nabla^2 h \tag{2.1}$$

where $z = h(x)$ is the height of the liquid surface, $z = b(x)$ is the height of the ground, $d(x) = h(x) - b(x)$ is the water depth, and g is gravitational acceleration. Waves have celerity $\sqrt{gd}$.

Equation (2.1) is discretely approximated (in the x-direction) using finite differences by

$$\frac{\partial^2 h_i}{\partial t^2} = -g\left(\frac{d_{i-1} + d_i}{2\left(\Delta x\right)^2}\right)(h_i - h_{i-1}) + g\left(\frac{d_i + d_{i+1}}{2\left(\Delta x\right)^2}\right)(h_{i+1} - h_i)$$

and numerically integrated using a first order implicit method where

$$\ddot{h}^n = \frac{\dot{h}^n - \dot{h}^{n-1}}{\Delta t}$$

$$\dot{h}^n = \frac{h^n - h^{n-1}}{\Delta t}$$

giving a system of linear equations with a simple tridiagonal matrix form that can be quickly solved. The linear nature of the problem and the implicit integration scheme make the solution method quite stable. The two-dimensional problem is solved by alternately solving the one-dimensional problem in the x and y directions using the alternating direction implicit (ADI) method.

2.2 Ship Wakes

Ship wakes—actually the foam and spray of ship wakes—have been simulated in real time using standard particle systems (Goss, 1990). New particles are periodically generated with initial positions and velocities determined by ad hoc kinematic and stochastic models of the bow and stern wakes.

Khan (1994) is modeling ship wakes by developing a function that describes the the shape of the water surface is response to ship position, direction, and speed. This kinematic function is based on fluid dynamic models.

2.3 Waterfalls

Sims (1990) animates a waterfall and other liquid phenomena by using thousands of motion-blurred particles, point masses that move under the influence of gravity. Efficiency is achieved by the use of a parallel implementation of Reeves's (1983) particle systems. The particles do not interact and can pass through each other.

2.4 Highly Viscous Liquids

Highly viscous liquids have been modeled as a collection of interacting particles (Miller and Pearce, 1989; Terzopoulos et al., 1989; Tonnesen, 1991). By simulating molecular forces between pairs of particles, the system of particles as a whole behaves like a highly viscous liquid, e.g., lava, mud, and slime (Miller and Pearce, 1989). In general the simulated force $\mathbf{f}_{ij}$ between particle i at position $\mathbf{x}_i$ and particle j at position $\mathbf{x}_j$ is repulsive at shorter ranges and attractive at longer ranges for surface tension (cf. section 3.6). In particular, Miller and Pearce (1989) used

$$\mathbf{f}_{ij} = (\mathbf{x}_i - \mathbf{x}_j)\left[s_r\left(\frac{\beta}{d^n} - \frac{\alpha}{d^m}\right)\right],$$

whereas Terzopoulos et al. (1989) used

$$\mathbf{f}_{ij} = m_i m_j \left(\mathbf{x}_i - \mathbf{x}_j\right) \left(\frac{\beta}{d^n} - \frac{\alpha}{(d+\zeta)^m}\right),$$

where ζ is how close the particles are allowed to be. Tonnesen (1991) incorporated thermal energy θ as

$$\mathbf{f}_{ij} = \left(\mathbf{x}_i - \mathbf{x}_j\right) \frac{-\epsilon_m + \phi}{n - m} \left(nm\frac{d_0^n}{d^{n+1}} - nm\frac{d_0^m}{d^{m+1}}\right),$$

where $d = |\mathbf{x}_i - \mathbf{x}_j|$ is the distance between the particles, α and β are constants of proportionality for the attractive and repulsive forces, and typically $n = 12$ and $m = 6$. Damping is added by the use of a force that is a function of the velocity and opposite in its direction.

While the straightforward algorithm for computing the interparticle forces is $O(n^2)$ for n particles (the force on each particle depends on all other particles), the range of influence of the interparticle force can be limited and by using spatial subdivision methods, the computational complexity of this approach can be reduced. Time-evolving dynamic equations of interacting point masses using a purely explicit numerical solution methods is more appropriate for "*soft* collisions" between particles, since in other cases stability problems emerge unless very small time steps are used (Miller and Pearce, 1989). Attaching field functions to these particles and rendering an isosurface of this field (blobbies) provides a way to model a liquid surface with changing topology.

2.5 State of the Art

To summarize the state of the art in modeling fluids for computer animation:

- Stochastic models of waves are relatively advanced and convincing and can be superimposed (via bump mapping) onto surface models controlled by kinematic and dynamic models for small-scale complexity.

- Kinematic models of waves are also quite convincing, and are usually efficient and easy to control, but require specific models for specific motions.
- Dynamic models describe motion at a higher level (in terms of forces—not velocity or position, both of which, of course, can be derived from forces) and hence are more general, but are hard to control, and can be inefficient, unstable, and inaccurate.
- Most wave models do not account for the changing topology of breaking waves. However, using particles as the control points for an implicit surface easily handles topological changes in the isosurface.
- Most models are ad hoc and difficult to integrate to describe more general fluid phenomena.

Going back a few centuries to the first principles of fluid dynamics for underlying and unifying concepts of fluid flow will address some of these difficulties.

Chapter Three

A Few Fundamentals of Fluid Flow

SECRET SACRED WARS ROACH: So many forces at work...so many cross-overs...so many tie-ins...I need more character!

FLEAGLE ROACHES: Uh-oh.

SECRET SACRED WARS ROACH: And we all know the only way to create character...don't we?

—from Dave Sim's Cerebus

This chapter briefly reviews some basic concepts of fluid mechanics necessary for the interactive flow field modeling methods of the next chapter.

3.1 Introduction

Fluid flow has been studied for centuries. The challenge is to select an appropriate framework for computer animation purposes, i.e, one that satisfies the accuracy, efficiency, controllability, and generality criteria defined in the first chapter. An early consideration is whether to look to molecular dynamics or to hydrodynamics. As mentioned in the previous chapter, point masses governed by molecular forces have been used to model viscous liquids (Miller and Pearce, 1989; Terzopoulos et al., 1989; Tonnesen, 1991). However, at a macroscopic scale, the scale of hydrodynamics and human perception, a liquid is considered to be continuous and essentially incompressible; at a microscopic scale, the scale of molecular dynamics, compressible behaviour can be observed, as molecules attract and repel one another. Conservation of liquid volume is not guaranteed. Also, it is unclear what constitutes the liquid boundary when simulating a liquid with a relatively small number of point masses. A macroscopic description of fluid, i.e., a hydrodynamic model, seems more appropriate than a model based on dynamics at the molecular level as it inherently alleviates most of the above problems.

To establish a background for the framework for integrating flow models for computer animation, the fundamentals of fluid mechanics are briefly reviewed[1].

3.2 Eulerian and Lagrangian Frames

In the field of hydrodynamics, the motion of liquids is described in two ways: Lagrangian and Eulerian. In the Lagrangian approach, the motion of each fluid particle is followed as it moves through space. Fluid velocity in a Lagrangian description $\mathbf{v}_i(t) = u\hat{\mathbf{i}} + v\hat{\mathbf{j}} + w\hat{\mathbf{k}}$ ($\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$, and $\hat{\mathbf{k}}$ are the unit vectors and t is time) is given in terms of the velocity of a particular particle i, and fluid acceleration is simply the change in the particle's velocity over time.

In the Eulerian approach, the motion of fluid particles at a given point in space is described. A Eulerian description of fluid velocity in Cartesian coordinates is given by:

$$\mathbf{v}(x,y,z,t) = u(x,y,z,t)\hat{\mathbf{i}} + v(x,y,z,t)\hat{\mathbf{j}} + w(x,y,z,t)\hat{\mathbf{k}}.$$

The acceleration in Eulerian terms is not simply the change in fluid velocity over time *at* a given point, but rather the change in fluid velocity of *a fluid particle moving through* a given point. It is given by the Stokes (or substantial or total) derivative of the velocity:

$$\frac{D\mathbf{v}}{Dt} = \frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v}\cdot\nabla)\,\mathbf{v} = \frac{\partial \mathbf{v}}{\partial t} + u\frac{\partial \mathbf{v}}{\partial x} + v\frac{\partial \mathbf{v}}{\partial y} + w\frac{\partial \mathbf{v}}{\partial z}.$$

Note that if a fluid particle is moving in a circle, for example, $\partial\mathbf{v}/\partial t = 0$ whereas $D\mathbf{v}/Dt$ is nonzero. An Eulerian frame is generally more convenient for work in fluid mechanics. An Eulerian description of liquid motion can be converted to Lagrangian terms by taking the Taylor series expansion of the Lagrangian velocity, which to first order is

$$\mathbf{v}_{\mathbf{a}}(t) = \mathbf{v}_E(\mathbf{a},t) + \left(\int_0^t \mathbf{v}_{\mathbf{a}}(t)dt\right)\cdot\nabla_{\mathbf{a}}\mathbf{v}_E(\mathbf{a},t).$$

[1]The interested reader is referred to the following books for a more detailed treatment of hydrodynamics: Crapper (1984), Feynmann et al. (1964), Kinsman (1984), Lamb (1945), Newman (1977), O'Neill (1986), Potter (1975), Rouse (1959), Roy (1988), Sabersky (1964), and Stoker (1987).

where $\mathbf{v_a}$ is the Lagrangian velocity of a fluid particle at point $\mathbf{a}$ at time $t = 0$ and $\mathbf{v}_E$ is the Eulerian velocity.

3.3 Equations of Motion

While the equations of motion for fluids may be complex, they are simply versions of Newton's second law of motion, as stated in the *Principia*:

> *the change of motion is proportional to the motive force impressed and is made in the direction of the right line in which that force is impressed,*

or

$$\mathbf{f} = m\mathbf{a},$$

where the mass m is the constant of proportionality. Note that the equations of motion express conservation of momentum: the change in momentum over time equals the applied forces.

The well-known Navier-Stokes equation is:

$$\rho \left(\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\, \mathbf{v} \right) = \rho \mathbf{f} - \nabla p - \frac{2}{3} \nabla \left(\mu \nabla \cdot \mathbf{v} \right),$$

where ρ is the density, p is the pressure, $\mathbf{f}$ is the external force including gravity, and μ is the coefficient of dynamic viscosity. The Navier-Stokes equation can be simplified by assuming the fluid is *inviscid* (or *ideal*), i.e., the viscosity, the friction between fluid particles, is zero. The velocity of a fluid next to a boundary is zero: notice the dust on the blades of a fan. The velocity just a short distance away can be quite large, and the viscosity can have a dominate effect on the liquid behaviour at this *boundary layer*, creating vorticity which, depending on the geometry and *Reynolds number* (ratio of inertial to viscous forces), may cause a negligible disturbance or may expand until it drastically affects the entire flow pattern causing *turbulent flow* where the velocity at a given point varies erratically. Turbulent flow is usually contrasted with *laminar flow* where the flow is steady near boundaries. Away from the boundary layer the effects of viscosity are generally negligible, and in the study of water waves, viscosity is usually ignored, i.e., the liquid is assumed to be inviscid.

If the fluid is assumed to be inviscid, then the Navier-Stokes equations of motion reduce to the Euler equations of motion, which in Eulerian terms are

$$\left.\begin{aligned} u_t + uu_x + vu_y + wu_z &= -\tfrac{1}{\rho}p_x \\ v_t + uv_x + vv_y + wv_z &= -\tfrac{1}{\rho}p_x - g \\ w_t + uw_x + vw_y + ww_z &= -\tfrac{1}{\rho}p_x \end{aligned}\right\} \tag{3.1}$$

or in vector form

$$\rho\frac{D\mathbf{v}}{Dt} = -\nabla p - \rho\mathbf{g} \ . \tag{3.2}$$

In Lagrangian terms, the Euler equations of motion are

$$\left.\begin{aligned} \rho a_x &= -p_x \\ \rho a_y &= -p_y - g \\ \rho a_z &= -p_z \end{aligned}\right\} \tag{3.3}$$

or in vector form

$$\rho\mathbf{a} = \rho\frac{d\mathbf{v}}{dt} = -\nabla p - \rho\mathbf{g} \tag{3.4}$$

where $\mathbf{v}(t)$ is the velocity in Lagrangian terms.

3.4 Kinematic Boundary Conditions

Fluid does not flow through solid boundaries, of course; this constraint is express by the kinematic boundary condition. For inviscid flow, if the fluid is in contact with a rigid boundary, the component of the velocity in the direction of the normal out of the fluid must match the normal component of the velocity of the boundary, $\mathbf{u}$,

$$\mathbf{v}\cdot\mathbf{n} = \mathbf{u}\cdot\mathbf{n}. \tag{3.5}$$

For a fixed boundary this reduces to:

$$\mathbf{v} \cdot \mathbf{n} = 0.$$

For viscous flow, there is a *no-slip* condition on the tangential component of the fluid velocity.

3.5 Bernoulli Equation

Often the Euler equations of motion (3.2) are more useful when they are integrated to give a form of Bernoulli's equation. Here we derive Bernoulli's equation for irrotational and possibly unsteady flow (as opposed to the other, perhaps more familiar forms of Bernoulli's equation for steady flow).

Euler's equation can be written as

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\,\mathbf{v} = -\frac{\nabla p}{\rho} - \mathbf{g}. \tag{3.6}$$

Defining the fluid speed as

$$v = |\mathbf{v}| = \sqrt{\frac{\partial \phi}{\partial x} + \frac{\partial \phi}{\partial y} + \frac{\partial \phi}{\partial z}},$$

where $\mathbf{v} = \nabla \phi$ and using the vector analysis identity

$$\nabla\left(\frac{1}{2}v^2\right) = \mathbf{v} \times (\nabla \times \mathbf{v}) + (\mathbf{v} \cdot \nabla)\,\mathbf{v},$$

allows the left-hand side of equation (3.6) to be written as

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\,\mathbf{v} = \frac{\partial}{\partial t}(\nabla \phi) + \nabla\left(\frac{1}{2}v^2\right) - \mathbf{v} \times (\nabla \times \mathbf{v})\,. \tag{3.7}$$

Since the fluid is assumed to be irrotational,

$$\nabla \times \mathbf{v} = 0,$$

and since $-gz = \mathbf{g}$, equation (3.7) can be reduced to

$$\nabla\left(\frac{\partial\phi}{\partial t} + \frac{1}{2}v^2 + \frac{p}{\rho} + gz\right) = 0.$$

Integration gives Bernoulli's equation for irrotational but possibly unsteady flow

$$\frac{\partial\phi}{\partial t} + \frac{1}{2}v^2 + \frac{p}{\rho} + gz = C(t). \tag{3.8}$$

As the constant of integration $C(t)$ is a function of time only and has no effect on any spatial gradient it is usually taken to be zero.

Often equation (3.8) is linearized and applied at the fixed surface $z = 0$:

$$\frac{\partial\phi}{\partial t} + g\eta + \frac{p}{\rho} = 0 \qquad \text{at } z = 0 \tag{3.9}$$

where $z = \eta(x, y, t)$ is the height of the free surface.

3.6 Surface Tension

Molecules throughout a liquid experience attractive molecular forces, but these forces cancel out in the interior of a liquid. On the surface of a liquid, however, a molecule experiences a net attractive force towards the interior of the liquid. Thus, molecules in the surface layer have a higher potential energy than molecules in the interior of the liquid, and since a liquid tends to minimize its potential energy, it tends to minimize its surface area. The resultant behaviour—as if the surface were a stretched membrane—produces what is called *surface tension.* Surface tension is typically modeled by constraining the curvature of the liquid surface.

3.7 Equation of Continuity

While the equations of motions express conservation of momentum, they do not express the conservation of mass. Considering the conservation of mass gives rise to another equation, the *continuity equation.* The continuity equation states the divergence of the density flux must be equal to the time change in density,

$$\nabla \cdot (\rho \mathbf{v}) = -\frac{\partial \rho}{\partial t}.$$

This equation can be simplified by assuming the fluid is *incompressible*, i.e. the fluid volume does not change with changes in applied pressure. This approximation is often made in the study of fluid flow, and is quite reasonable for computer animation purposes. For example, the density of water increases by no more than 0.5% when the pressure applied is increased by 100 atmospheres for constant temperatures (Stoker, 1957). And while gaseous flow may seem to be compressible flow, almost all examples of compressible behaviour of a gas occur with respect to a closed volume—when the gas is free to move, it simply moves without being compressed (Potter, 1975).

The assumption that the fluid is incompressible allows the continuity equation to be simplified to

$$\nabla \cdot \mathbf{v} = 0. \tag{3.10}$$

Thus, perhaps surprisingly, conservation of mass is expressed without a term for mass. For constant density[2], the continuity equation also expresses conservation of fluid volume. Any surface advected with the flow has no flow through it and conserves the mass of the domain it bounds. The vector form of the continuity equation can be written out in in three common coordinate systems that will prove convenient: Cartesian, cylindrical and spherical. In Cartesian

[2]Note that technically *incompressible* does not necessarily mean the density is constant. A necessary and sufficient condition for a fluid to be incompressible is that the Stokes derivative of the liquid density be zero, i.e., $D\rho/Dt = 0$, so while the density is constant along streamlines, it is not necessarily constant at all points of the fluid. However, this distinction is often ignored.

coordinates the continuity equation is:

$$\frac{\partial v_x(x,y,z)}{\partial x} + \frac{\partial v_y(x,y,z)}{\partial y} + \frac{\partial v_z(x,y,z)}{\partial z} = 0 \tag{3.11}$$

In cylindrical coordinates the continuity equation is:

$$\frac{\partial (r v_r(r,\theta,z))}{r\partial r} + \frac{\partial v_\theta(r,\theta,z)}{r\partial \theta} + \frac{\partial v_z(r,\theta,z)}{\partial z} = 0 \tag{3.12}$$

In spherical coordinates the continuity equation is:

$$\frac{\partial (v_R(R,\theta,\phi)R^2)}{R^2\partial R} + \frac{\partial (v_\theta(R,\theta,\phi)\sin\theta)}{R\sin\theta\partial\theta} + \frac{\partial v_\phi(R,\theta,\phi)}{R\sin\theta\partial\phi} = 0 \tag{3.13}$$

For irrotational flow, i.e., when the curl of the velocity is zero, the velocity field can be represented as the gradient of a scalar potential field:

$$\nabla\phi(x,y,z,t) = \mathbf{v}(x,y,z,t)$$

which when substituted into the continuity equation for incompressible fluid flow gives the potential (Laplace) equation:

$$\nabla^2\phi(x,y,z) = 0.$$

Chapter Four

Interactive Flow Field Modeling

in-flu-ence [ME, fr. MF, fr. ML influentia, fr. L influent-, influens, prp. of influere to flow in, fr. in- + fluere to flow – more at FLUID] 1a: an ethereal fluid held to flow from the stars and to affect the actions of humans

—Webster's 7th Dictionary

While dynamic simulation of fluid motion can be done efficiently enough for computer animation purposes under restrictive conditions (Kass and Miller, 1990), in general dynamic simulation of fluid flow appears to be too computationally expensive to be appropriate as a general tool for realistically animating fluids. Using straight dynamic simulation of hydrodynamic models as a method for animating fluids begs the following problems:

1. How are initial conditions realistically modeled?

2. How are turbulent effects created and controlled?

3. How is the motion controlled in general?

In this chapter interactive flow field modeling methods are presented to address these problems.

4.1 Overview

Fluids present several challenges for computer animation stemming from their continuously deformable nature. To date, these challenges have not fully addressed by dynamic simulation methods. Consider using dynamic simulation to animate water flowing out of hose, making a loop, and flowing into a bucket. Hydrodynamic simulation can describe fluid motion in response to the forces acting on the fluid. These forces can be modeled (Sims, 1990), but would require the animator to guess what forces are required to have the water make a loop and flow into the bucket. While the dynamic simulation of articulated solid objects can be controlled using kinematic constraints and inverse dynamics (Barzel and Barr, 1988; Isaacs and Cohen, 1987; Platt and Barr, 1988; Witkin and Kass, 1988), with fluids the number of degrees of freedom is essentially infinite, so these constraint methods do not appear to be applicable. A priori, flow field modeling appears more appropriate. Sims (1990) presented ad hoc "velocity operators" for controlling the motion of particle systems. Ebert et al. (1993) used a combination of three-dimensional grids and ad hoc flow field functions to model flow fields. In these methods, the task of ensuring realism is left to the animator. A more accurate method is to use superposition of simple solutions to the potential equation such as those corresponding to source, sink, and vortex flow to model more complex flow fields (Haumann, 1991a; Haumann, 1991b; Wejchert and Haumann, 1991; Haumman and Hodgins, 1992).

Using just these point source, sink, and vortex "flow primitives", however, makes certain modeling tasks difficult, e.g., modeling a path for the fluid to follow. To allow more control in modeling flow fields, a flow primitive based on a cubic B-spline is introduced. A new method of modeling turbulent flow is presented that gives the animator greater control over the turbulence. Finally, a simple method of blending interactively modeled and dynamically simulated flow fields is presented.

4.2 Principle of Superposition

The point flow primitives, the spline-based flow primitives, the turbulence modeling, and the contol method for dynamic simulation are all integrated by a common framework: the continuity equation for incompressible flow (3.10). For partial differential equations that are both linear and homogeneous, such as equation (3.10), the *principle of superposition* applies: if $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ are solutions to equation (3.10) and $c_1, c_2, \ldots, c_n$ are scalars, then a linear combination of these solutions

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_n\mathbf{v}_n$$

is also a solution. Thus, flow fields sastifying the continuity equation for incompressible flow can be arbitrarily scaled and summed and result in a flow field that satisfies the continuity equation. This principle is the basis of all the methods in this chapter.

4.3 Basic Flow Fields

A number of simple flow fields that satisfy the continuity equation for incompressible flow are well-known. Such flow fields include those corresponding to uniform, point source, point sink, point dipole, and continuous straight-line vortex flows. A well-known method of constructing more complex flow fields is to superpose a number of these "basic" flow fields.

4.3.1 Uniform

The simplest flow pattern is *uniform flow* where the fluid velocity at every point is the same, $a\hat{\mathbf{i}} + b\hat{\mathbf{j}} + c\hat{\mathbf{k}}$. This clearly satisfies the continuity equation for incompressible flow:

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} = \frac{\partial a}{\partial x} + \frac{\partial b}{\partial y} + \frac{\partial c}{\partial z} = 0$$

Note that a linear combination of uniform flow fields will simply result in another uniform flow field.

More generally,

$$\mathbf{v} = (a + f(y,z))\hat{\mathbf{i}} + (b + f(x,z))\hat{\mathbf{j}} + (c + f(x,y))\hat{\mathbf{k}}$$

satisfies the continuity equation for incompressible flow.

4.3.2 Point Source and Sink

Another basic flow field is one corresponding to the flow from a point source located at the origin (Figure 4.1).

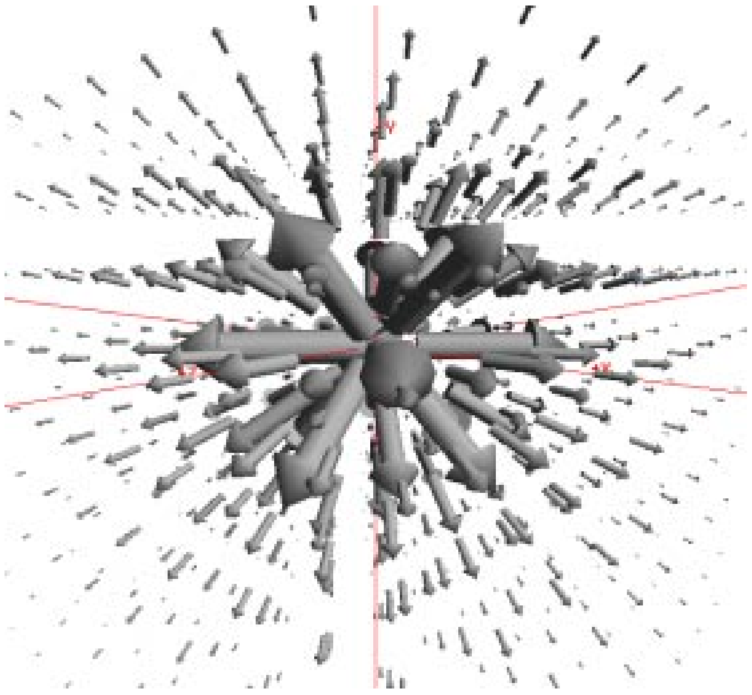

Figure 4.1: Flow field of point source.

In spherical coordinates (R, θ, ϕ) this flow field is given by:

$$\mathbf{v} = \frac{m}{R^2}\hat{\mathbf{i}}. \tag{4.1}$$

Note that there is a singularity at the origin. The constant m is called the strength of the source: the discharge through any closed surface surrounding the origin is $4\pi m$.

It can be easily seen that equation (4.1) sastifies the continuity equation by recalling the continuity equation in spherical coordinates:

$$\frac{\partial(uR^2)}{R^2\partial R} + \frac{\partial(v\sin\theta)}{R\sin\theta\partial\theta} + \frac{\partial w}{R\sin\theta\partial\phi} = 0.$$

Negating equation (4.1) gives

$$\mathbf{v} = -\frac{m}{R^2}\hat{\mathbf{i}},$$

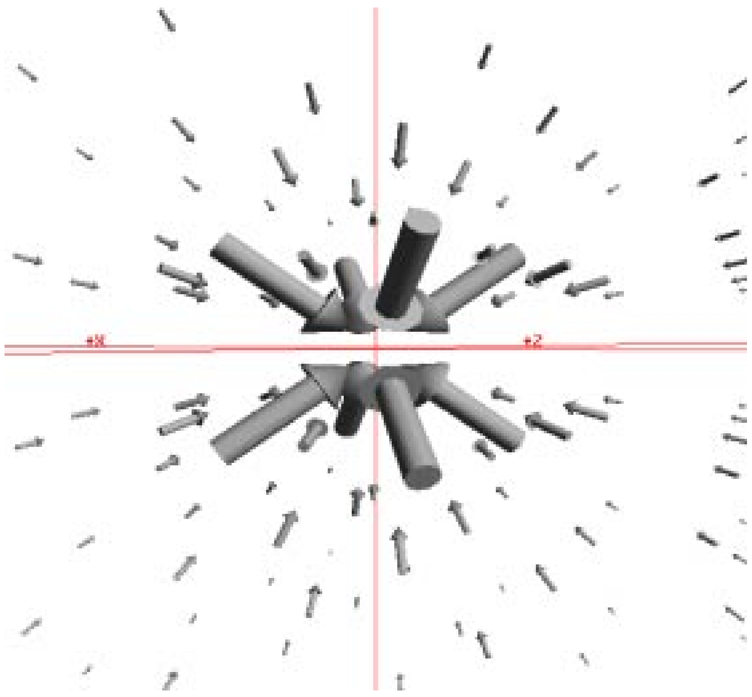

Figure 4.2: Flow field of point sink.

which represents the flow towards a point sink at the origin (Figure 4.2).

4.3.3 Vortex

Using cylindrical coordinates (r, θ, z), the flow field corresponding to straight-line vortex is given by:

$$\mathbf{v} = \frac{k}{r}\hat{\mathbf{j}}. \tag{4.2}$$

Figure 4.3 shows the flow field for a vortex.

4.4 Spline-Based Flow Field Modeling

While these basic flow fields (point flow primitives) can be linearly combined to produce more complex flow fields that also satisfy the continuity equation for incompressible flow, it is difficult

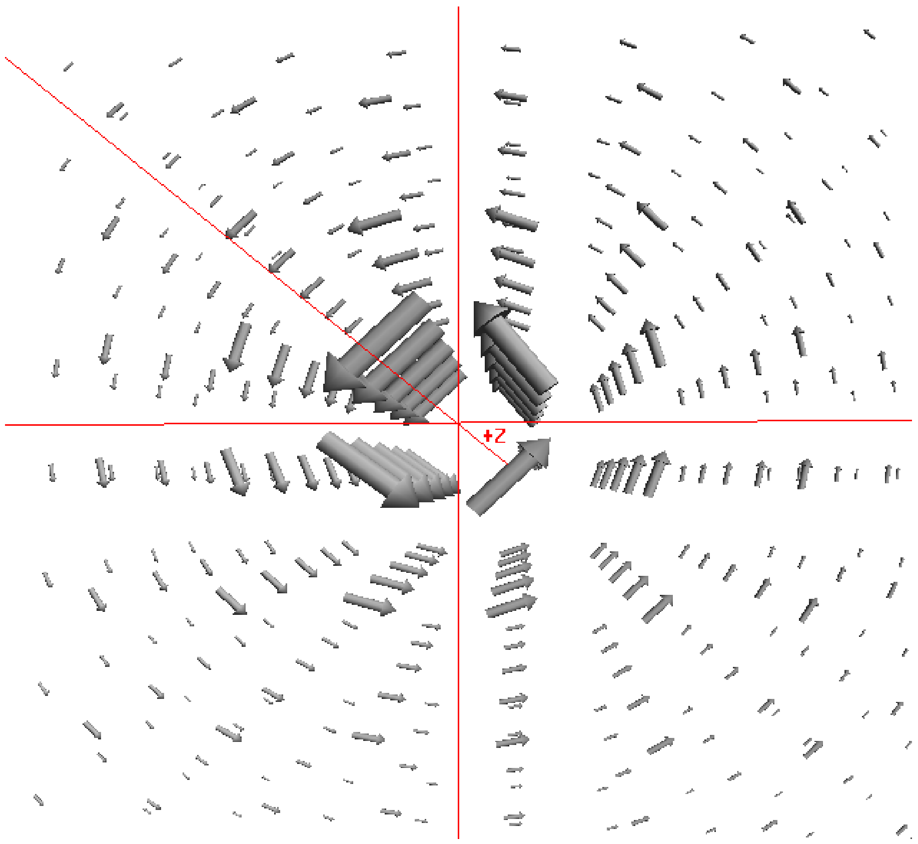

Figure 4.3: Flow field of line vortex.

to obtain specific motion, e.g., directed motion along a given path, using only these basic flow fields. A more powerful flow primitive is desired.

4.4.1 Overview

An interactive flow field modeling method should minimize the effort required by the animator to realize the desired flow field. An intuitive approach would be for the animator to design the general flow pattern by specifying some streamlines in the flow field. Streamlines are lines that are tangential to the fluid velocity; for steady flow, they are the lines a fluid particle will follow. The method would then automatically compute a realistic flow field that contains those streamlines. This computation could be accomplished by a relaxation method or some other numerical solution method given that a well-defined problem could be developed. However, such a method would require solving a partial differential equation over three dimensions which probably would not be fast enough for interactive work on standard graphics workstations. A simpler and much more efficient approach is to use flow primitives based on curves, say cubic B-splines, that have flow fields where the streamlines follow the shape of the curve and attenuate in magnitude away from the spline. Superposition of these spline-based flow primitives would then automatically give blending between the primitives where each primitive could be weighted for either local or global effect.

As will be shown, such a spline-based flow primitive can be developed by integrating a special point primitive along a spline. This point primitive—call it a *directed flow primitive*—has a specified velocity at a given point (say the origin) that attenuates in magnitude moving away from this point yet maintains the same direction as much as possible while satisfying the continuity equation. This directed flow primitive can then be continuously distributed (integrated) over a spline such that the direction of each directed flow primitive on the spline is always tangential to the spline, giving a flow field where the streamlines of the flow correspond to the shape of the spline and attenuate radially in magnitude away from the spline.

4.4.2 A Directed Flow Primitive

More specifically, the directed flow primitive should have a flow field with the following properties:

- The velocity is a maximum at the origin and is oriented in the direction of the positive z axis.
- The magnitude of the velocity goes to zero moving away from the origin.
- The direction of the velocity everywhere is close to the direction at the origin, more so where the magnitude of the velocity is greater.
- The magnitude of z coordinate of the velocity can be an arbitrary function of the radial distance from the z axis.
- The flow field satisfies the continuity equation.

Together, these properties ensure when this flow field function is integrated along a curve, the streamlines of the resulting flow field correspond to the shape of the curve and the flow is strongest near the curve.

Such a flow primitive was derived in cylindrical coordinates (Figure 4.4):

$$\mathbf{v} = v_r \hat{\mathbf{i}} + v_\theta \hat{\mathbf{j}} + v_z \hat{\mathbf{k}} \tag{4.3}$$

where

$$v_r(x, y, z, t) = \frac{cz}{(a + z^2)^2 (b + r^2)}, \tag{4.4}$$

$$v_\theta(x, y, z, t) = 0, \tag{4.5}$$

and

$$v_z(x, y, z, t) = \frac{c}{2(a + z^2)} \left(\frac{1}{r(b + r^2)} - \frac{2r}{(b + r^2)^2} \right) + f(r) \tag{4.6}$$

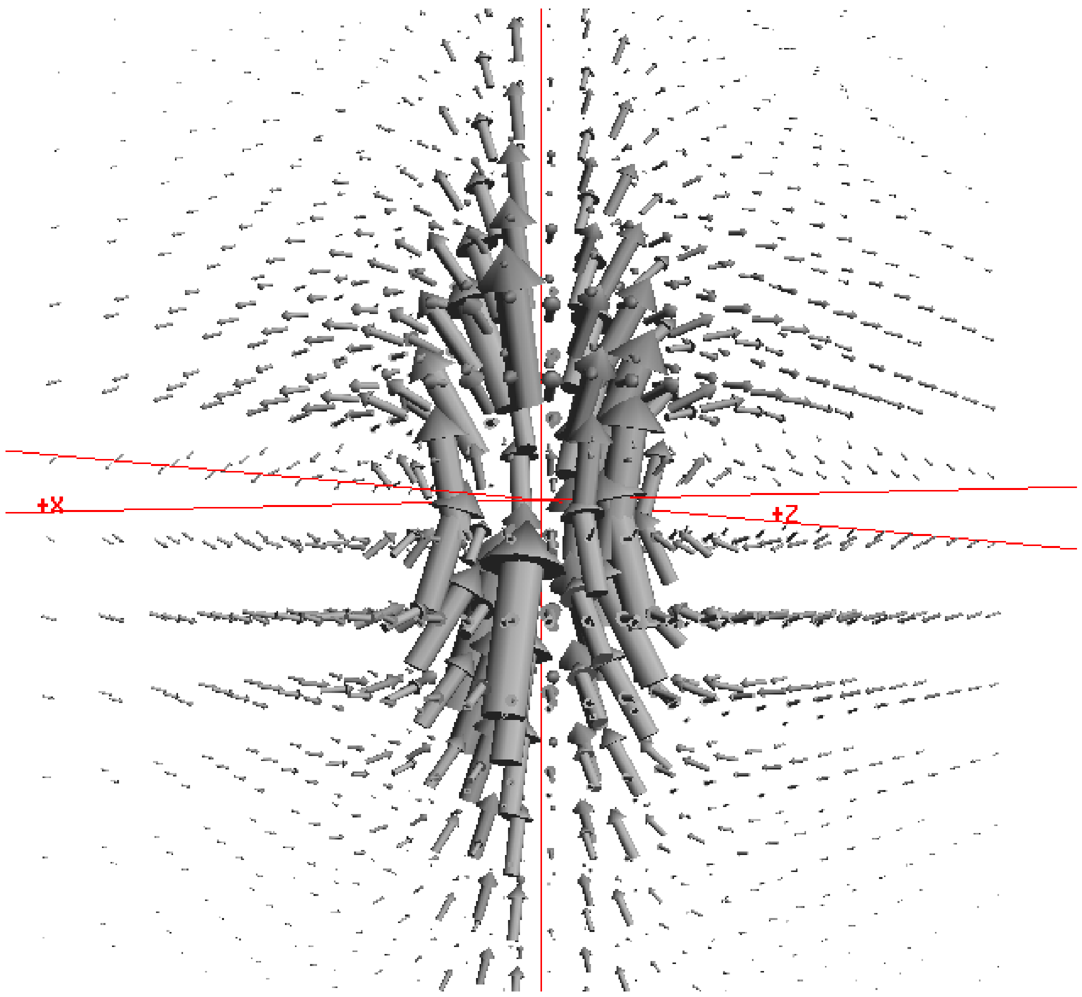

Figure 4.4: Flow field for integration over spline.

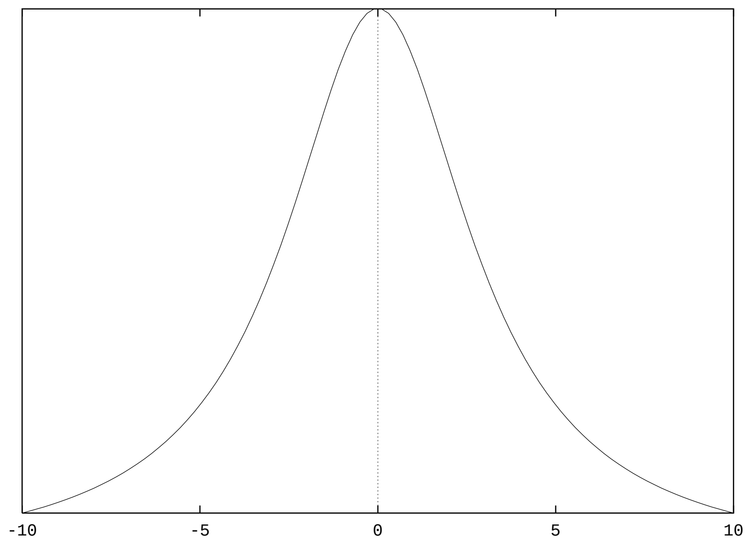

Figure 4.5: A plot of the v_z component of the directed flow primitive for a fixed r.

where a, b, c are constants. Figure 4.5 shows v_z for a fixed r. A plot of v_r for a fixed z would be similar.

This flow field can be shown to satisfy the continuity equation which in cylindrical coordinates is:

$$\frac{\partial\left(r v_r(r,\theta,z)\right)}{r\partial r} + \frac{\partial v_\theta(r,\theta,z)}{r\partial\theta} + \frac{\partial v_z(r,\theta,z)}{\partial z} = 0 \tag{4.7}$$

as

$$\frac{\partial\left(r v_r\right)}{r\partial r} = \frac{\partial}{r\partial r}\left(\frac{crz}{\left(a+z^2\right)^2\left(b+r^2\right)}\right) = \frac{cz}{(a+z^2)^2}\left(\frac{1}{r(b+r^2)} - \frac{2r}{(b+r^2)^2}\right)$$

and

$$\frac{\partial v_z}{\partial z} = \frac{\partial}{\partial z}\left[\frac{c}{2(a+z^2)}\left(\frac{1}{r(b+r^2)} - \frac{2r}{(b+r^2)^2}\right) + f(r)\right] = \frac{-2cz}{2(a+z^2)^2}\left(\frac{1}{r(b+r^2)} - \frac{2r}{(b+r^2)^2}\right)$$

4.4.3 A Spline-Based Flow Primitive

In this section various flow field modeling "tools" are developed based on cubic B-splines. Cubic B-splines are chosen as a basis of a fluid motion modeling tool because of the popularity of cubic B-splines in motion modeling in general, are inherently smooth (C^2 continuous), have local control, and are available in common graphics software libraries such as Silicon Graphics Graphics Library$^{\text{TM}}$ which supports NURBS curves (Silicon Graphics, 1991).

Cubic B-splines are a type of parametric piecewise cubic curves that approximate a series of $m+1$ control points $P_0, P_1, \ldots, P_m, m \geq 3$, with a curve consisting of $m-2$ cubic polynomial segments $Q_3, Q_4, \ldots, Q_m$. The parameter range on which Q_i is defined is $t_i \leq t < t_{i+1}$ for $3 \leq i \leq m$. For each $i \geq 4$, there is a join point or knot between Q_{i-1} and Q_i at the parameter value t_i (the knot value). The initial point at t_3 and final point at t_{m+1} are also knots, for a total of $m-1$ knots.

The uniform nonrational B-spline formulation for curve segment i is

$$\mathbf{Q}_i(t) = \left[\begin{array}{ccc} x_i(t) & y_i(t) & z_i(t) \end{array}\right] = \mathbf{T}_i \mathbf{M}_{Bs} \mathbf{G}_{Bs_i} \quad t_i \leq t < t_{i+1} \tag{4.8}$$

where

$$\mathbf{T} = \left[\begin{array}{cccc} (t-t_i)^3 & (t-t_i)^2 & (t-t_i) & 1 \end{array}\right]$$

and the B-spline basis matrix is

$$\mathbf{M}_{Bs} = \frac{1}{6} \left[\begin{array}{cccc} -1 & 3 & 3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{array}\right] \tag{4.9}$$

and the B-spline geometry vector is:

$$\mathbf{G}_{B_{S_i}} = \begin{bmatrix} P_{i-3} \\ P_{i-2} \\ P_{i-1} \\ P_i \end{bmatrix} \tag{4.10}$$

Expanding equation (4.8) out gives:

$$\mathbf{Q}_i(t - t_i) = \frac{(1-t)^3}{6}P_{i-3} + \frac{3t^3 - 6t^2 + 4}{6}P_{i-2} + \frac{3t^3 + 3t^2 + 3t + 1}{6}P_{i-1} + \frac{t^3}{6}P_i \tag{4.11}$$

for $0 \leq t < 1$ and $3 \leq i \leq m$.

Nonuniform nonrational B-splines are given by:

$$\mathbf{Q}_i(t) = \mathbf{P}_{i-3}\mathbf{B}_{i-3,4} + \mathbf{P}_{i-2}\mathbf{B}_{i-2,4} + \mathbf{P}_{i-1}\mathbf{B}_{i-1,4} + \mathbf{P}_i\mathbf{B}_{i,4} \tag{4.12}$$

for $3 \leq i \leq m$ and $t_i \leq t < t_{i-1}$ where

$$B_{i,1}(t) = \begin{cases} 1 & t_i \leq t < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{4.13}$$

$$B_{i,2}(t) = \frac{t - t_i}{t_{i+1} - t_i}B_{i,1}(t) + \frac{t_{i+2} - t}{t_{i+2} - t_{i+1}}B_{i+1,1}(t) \tag{4.14}$$

$$B_{i,3}(t) = \frac{t - t_i}{t_{i+2} - t_i}B_{i,2}(t) + \frac{t_{i+3} - t}{t_{i+3} - t_{i+1}}B_{i+1,2}(t) \tag{4.15}$$

$$B_{i,4}(t) = \frac{t - t_i}{t_{i+3} - t_i}B_{i,3}(t) + \frac{t_{i+4} - t}{t_{i+4} - t_{i+1}}B_{i+1,3}(t) \tag{4.16}$$

Integration over a parametric curve C with parameter t and arc length s is given by:

$$\int_C f(t)\frac{ds}{dt}dt.$$

The the arc length for parametric curves in three dimensions s is

$$s = \int_a^b \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2}\, dt$$

and so integrating basic flow fields over splines takes the form:

$$\int_Q \mathbf{v}\,(x, y, z, a(t), b(t), c(t)) \sqrt{\left(\frac{da}{dt}\right)^2 + \left(\frac{db}{dt}\right)^2 + \left(\frac{dc}{dt}\right)^2}\, dt \tag{4.17}$$

Gaussian quadrature can be used to approximate the integral:

$$\int_a^b W(x) f(x) dx \approx \sum_{i=1}^{N} w_i f(x_i)$$

When $W(x) = 1$, the numerical integration procedure is known as Gauss-Legendre integration, and there are procedures to calculate the weights and abscissas (Press et al., 1988).

flow field correspond to the shape of the spline (Figures 4.6 and 4.4.3). Figure 4.8 shows the flow field resulting from two spline flow primitives.

When kinematic control over the *changes* in the flow field are desired, time-varying flow primitives can be used. The existence, strength, position, and orientation (when significant) of these primitives can vary in time. Note that the rate of change of the strength of the flow primitives is directly related to a force acting on the flow field. These changes can be made explicitly by the animator or a wide range of models can be used, e.g., the flow primitives could be advected by a flow field modeled by the methods of this chapter.

4.5 Turbulence

Turbulent flow produces much of the visual complexity of fluid motion. A number of computer graphics researchers have developed models of turbulent flow fields that can be used (Shinya and Fournier, 1992; Ebert, 1993; Stam and Fiume, 1993). These flow fields are modeled using

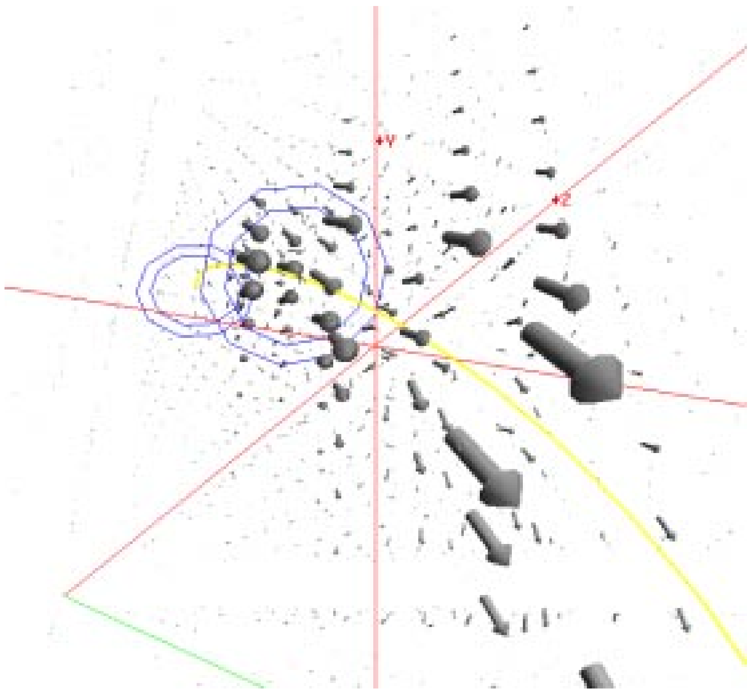

Figure 4.6: A spline flow primitive.

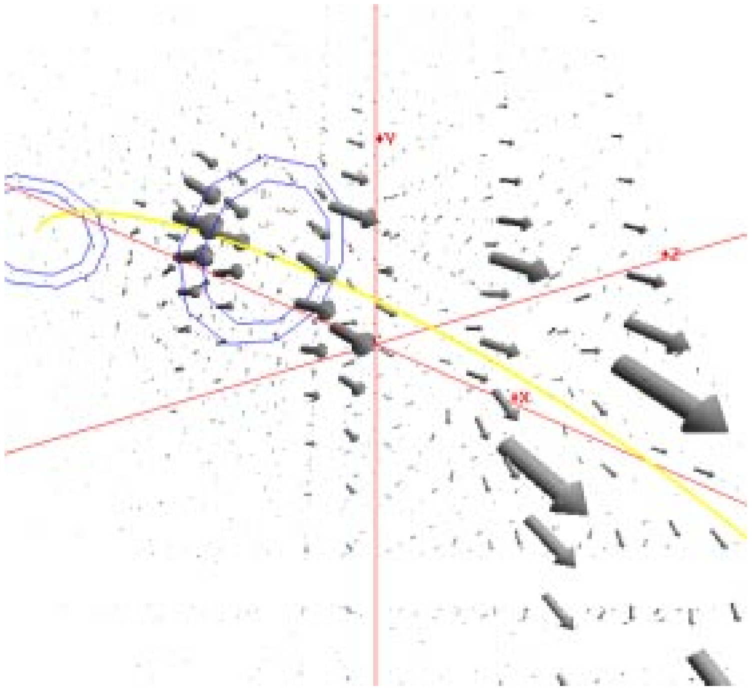

Figure 4.7: Another view of spline flow primitive.

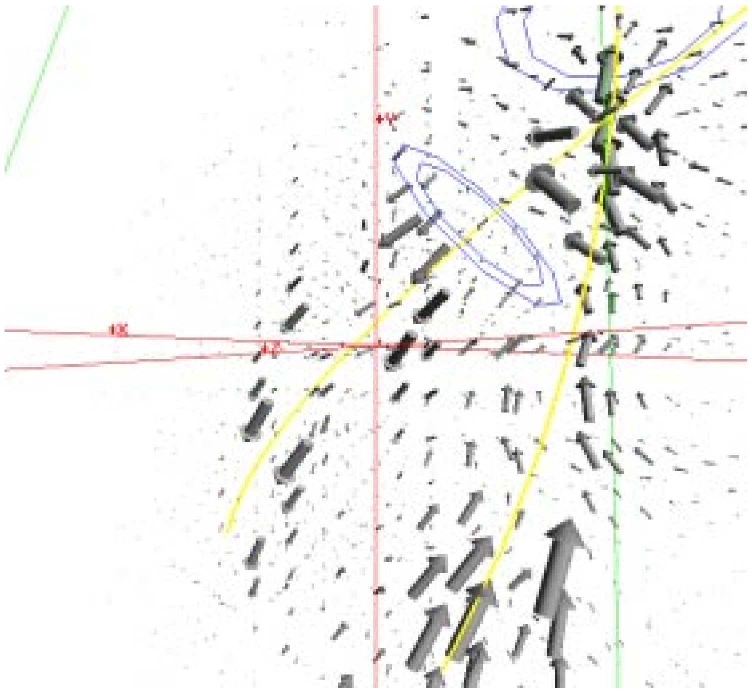

Figure 4.8: Flow field of two spline flow primitives.

Fourier synthesis, i.e., modifying a white noise spectrum in the frequency domain and then taking the inverse FFT of it. This approach presents several difficulties for animating the turbulent motion of fluids:

- The fluid has no effect on the turbulent field.
- It is not clear how to localize turbulent motion in space or time.
- Uniform grids are generally required, limiting resolution.

For example, it is unclear how to use these methods to create an animation of a rock in a river creating turbulent motion that diffuses downstream.

An alternative approach is suggested based on a concept used in fluid dynamics: describing turbulent flow in terms of vortices of various shapes, size, and rotational speeds being carried along by the mean flow (Rouse, 1959). Flows corresponding to vortex flow can be generated where stochastic motion is desired and these flows can be advected with the mean flow field. The

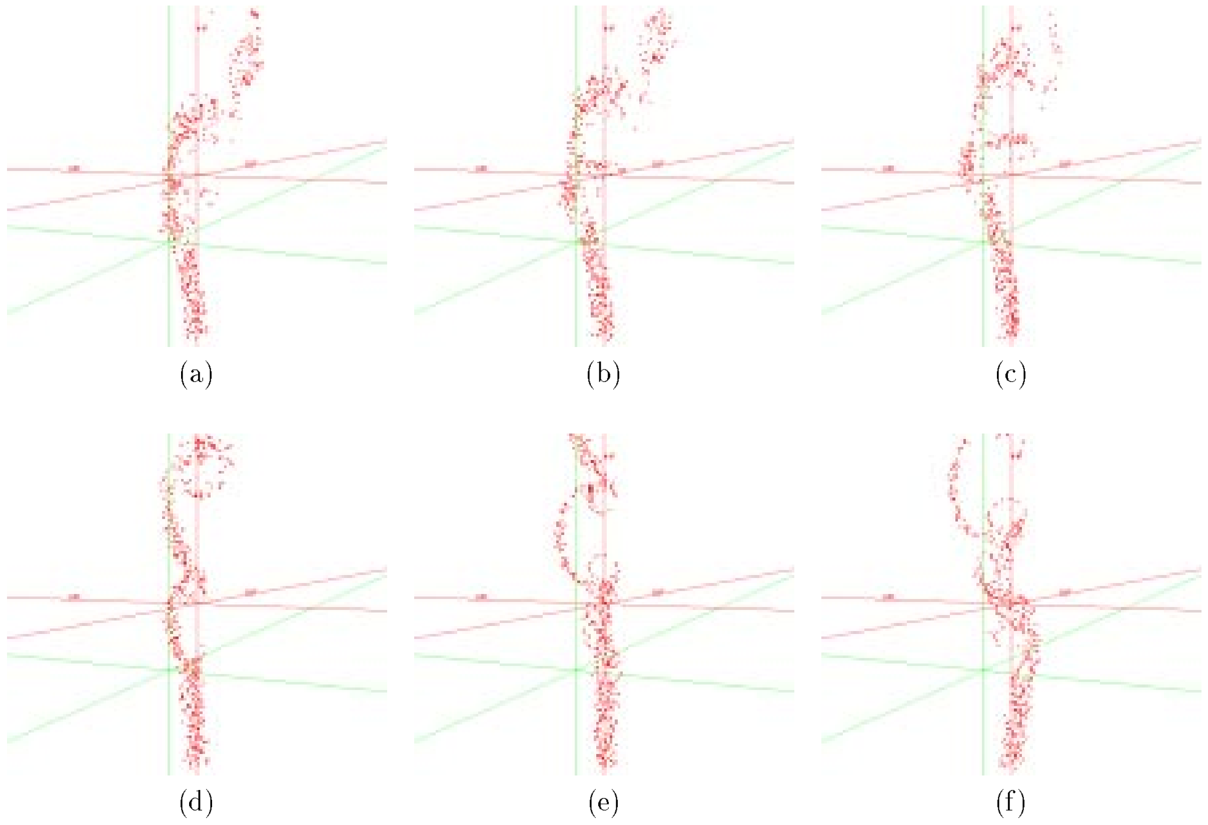

Figure 4.9: Turbulent flow.

stochastic distribution (in space and time) of these vortices can be based on power spectrums introduced in graphics or from the study of turbulent flow or be specified by the animator. Further, the attributes of these vortices can change over time, and the vortices themselves can eventually dissipate according to parameters defined by the animator. Besides interactive approaches to modeling the generation of turbulence, models of the formation of turbulence can be used. Turbulence can always safely be assumed to be present in flows previously or currently in a state of shear at moderate to high Reynolds numbers (ratio of inertial to viscous forces). Figure 4.9 shows frames from an interactive animation of turbulent flow. A uniform flow field is used as the mean flow field. Particles are advected upwards from a fluid source at the bottom of the frames. Part of the way up, a turbulent source generates vortices for a simulation of, for example, smoke from a recently extinguished candle.

4.6 A Continuum of Control

As discussed, the continuity equation for incompressible flow (3.10) can be used as a basis for the framework for integrating flow models for computer animation discussed in the first chapter. This equation completely and kinematically describes an incompressible flow field and is independent of the equations of motion being used.

In the framework, fluid flow is modeled in an Eulerian frame, and the flow is defined everywhere. The flow is decomposed into two main components, mean flow and turbulent flow. The mean flow is further decomposed into flows interactively modeled using kinematic methods and flows derived from dynamic simulation. The mean flow can be computed on a grid to make it more efficient; the turbulent flow is described by continuous functions for detail at small-scales.

Using dynamic models that describe changes in the flow field, such as the Navier-Stokes equations, dynamic simulation methods can be used to describe a time-dependent flow field $\mathbf{v}_d(x, y, z, t)$.

The flow primitives presented in this chapter can be arbitrarily superposed and linearly weighted in time as:

$$\sum_{i=1}^{n} c_i(t)\mathbf{v}_i(x, y, z, t)$$

where

$$\frac{dc_i}{dt} \tag{4.18}$$

is directly related to a force on the fluid, but unlike modeling forces, modeling the flow field allows the animator to deal directly with the motion of the fluid and not with the *changes* in the motion of the fluid.

And finally, turbulence can be modeled by using a turbulent flow field, $\mathbf{v}_t$. This flow field can be computed using the methods of Stam and Fiume (1993), Shinya and Fournier (1992), or the methods introduced in this chapter.

These flow fields can be trivially integrated by superposition:

$$\mathbf{v} = c_d \mathbf{v}_d(x, y, z, t) + \sum_{i=1}^{n} c_i(t)\mathbf{v}_i(x, y, z, t) + c_t \mathbf{v}_t(x, y, z, t) \tag{4.19}$$

while sastifying the continutiy equation for incompressible flow:

$$\nabla \cdot \left(c_d \mathbf{v}_d(x, y, z, t) + \sum_{i=1}^{n} c_i(t)\mathbf{v}_i(x, y, z, t) + c_t \mathbf{v}_t(x, y, z, t) \right) = 0. \tag{4.20}$$

Thus, the animator can arbitrarily weight the effects of the flow field from dynamic simulation and the interactively modeled flow field, providing a continuum of control from letting dynamic simulation run free to an explicitly modeled flow pattern. Further, turbulent motion can be superposed on the resulting flow field at will.

Chapter Five

Implementation and Results

Those are my principles. If you don't like them I have others.
—Groucho Marx

In this chapter, the implementation of a small animation system incorporating the interactive flow field modeling methods of the previous chapter is described. The use of this system and the flow modeling methods are discussed.

5.1 Implementation

In order to test the framework and flow modeling methods, it was necessary to implement a relatively small animation system (about 10,000 lines of code). While various toolkits exist to assist the development of animation systems, fluid flow does not mesh well with the currently popular object-oriented paradigm of such toolkits, and thus an animation system had to be written from scratch. Using the C programming language, the workhorse of the computer graphics community, and Silicon Graphics' Graphics Library (GL) [cite], the animation system FLOW was implemented. FLOW allows the interactive modeling and visualization of flow fields and provides a testbed for the methods of the previous chapter. Geometric models, from particles to complex polyhedrons, can be advected in the flow field. Animation scripts for the renders Rayshade [cite] and Optik [cite] can be automatically generated.

5.1.1 Interaction

A large part (roughly a third) of the implementation concerns the user interface. The interface toolkit FORMS (cite) greatly simplified this task.

The user can position flow primitives using the mouse to model the mean flow field and then place turbulent sources in this flow field. Geometric models can be interactively positioned in the flow field. Most modeling parameters can be interactively adjusted.

5.1.2 Flow Visualization

Visualization of vector fields is an active area of computer graphics research. Two standard techniques were found to be effective, though not ideal: particle traces and vector arrows. With particle traces (for steady flow, these are streamlines) the user can position a cursor in the flow field, and from a disc normal to the velocity at that point, particles are advected by means of Runge-Kutta numerical integration for an user-given distance. With vector arrows, a grid is constructed where the nodes have arrows representing the magnitude and direction of the velocity at that point.

5.1.3 Rendering

For interactive work, geometric primitives are advected with the flow using standard graphics library routines. Several thousand particles can be advected at interactive rates on a Silicon Graphics Crimson workstation. For producing the final animation, scripts are output that can be used by the raytracers Optik and Rayshade to produce high-quality images.

5.2 Using FLOW

To create an animation using FLOW, the following steps are taken:

1. Using point and spline flow primitives, a flow field is constructed.
2. Interactively using one of the flow the visualization methods, the flow field is inspected and modified.

3. Geometric models are interactively placed into the flow field and advected by it.

4. When a satisfactory sequence has been obtained, the results are recorded as a series of Optik or Rayshade scripts.

Flow fields from dynamic simulations can be imported into FLOW, but this has not been tested.

5.3 Results

Using cubic B-splines as flow primitives is a promising method of designing flow fields, but more work is required to refine this primitive to make it a useful animation tool. The number of degrees of freedom available to the animator is quite high: the animator is responsible for controlling the magnitude and radial influence of the flow along each spline. The turbulent flow modeling methods were surprisingly effective, and a small number of vortices were required to give good results. The simplicity of this method is attractive to the animator.

Flow around objects can be simulated by the using of point sources. A simple example is a single point source used to create flow around a sphere (see figure 5.1). In general a distribution of point sources and sinks can be used to simulate flow around an object. For example, in the plane, a source-sink pair is known as a Rankine oval as it gives the flow pattern around an oval-shaped object. Realizing accurate flow patterns around complex objects using distributions of point primitives is a nontrivial task. For computer animation, often crude approximations to the actual flow pattern are adequate.

A short animation called *Liquid Skull* was produced to demonstrate how a polygonal object can be advected in an animator-designed flow field (see figures 5.2 to 5.3). This animation represents X-rays of an overly ambitious graduate student's skull as he tackles a thesis topic that is much too large.

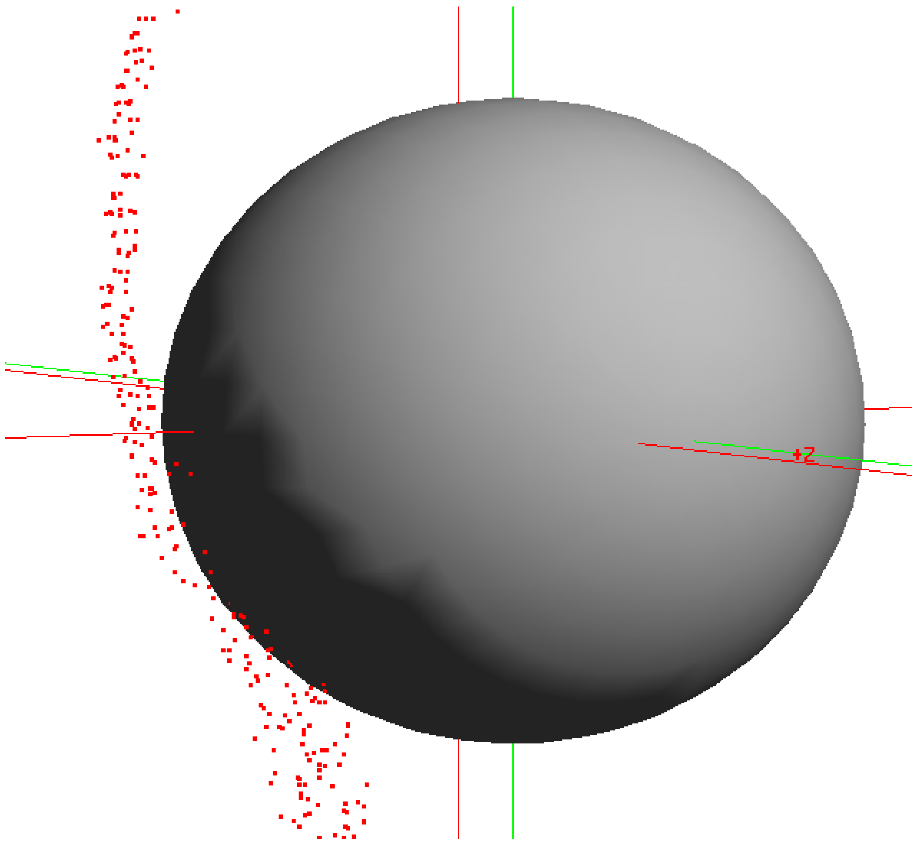

Figure 5.1: Flow around object.

Figure 5.2: Frame one from *Liquid Skull*

Figure 5.3: Frame two from *Liquid Skull*

Figure 5.4: Frame three from *Liquid Skull*

Figure 5.5: Frame four from *Liquid Skull*

Figure 5.6: Frame five from *Liquid Skull*

Figure 5.7: Frame six from *Liquid Skull*

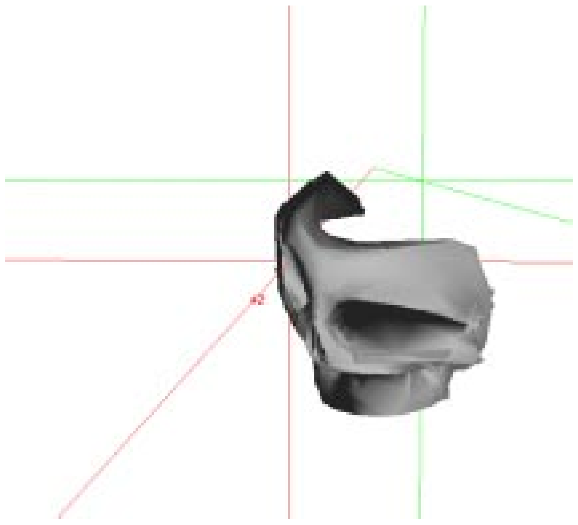

Figure 5.8: Frame seven from *Liquid Skull*

Figure 5.9: Frame eight from *Liquid Skull*

Figure 5.10: Frame nine from *Liquid Skull*

Figure 5.11: Frame ten from *Liquid Skull*

Figure 5.12: Frame eleven from *Liquid Skull*

Figure 5.13: Frame twelve from *Liquid Skull*

Figure 5.14: Frame thirteen from *Liquid Skull*

Figure 5.15: Frame fourteen from *Liquid Skull*

Figure 5.16: Frame fifteen from *Liquid Skull*

Figure 5.17: Frame sixteen from *Liquid Skull*

Chapter Six

Conclusions and Future Work

Don't let it end like this. Tell them I said something.
—Last words of Pancho Villa

6.1 Conclusions

A general framework based on first principles of fluid mechanics, in particular the conservation of fluid mass, was presented. This simple framework, using linear combinations of solutions to the continuity equation for incompressible flow, allows the realistic blending of animator-modeled flow fields and flow fields resulting from dynamic simulation. Numerical solution of even the simplest hydrodynamic models for the types of flows of interest to computer animation—three-dimensional, free-surface flow with changing topologies—remains an active area of computational fluid dynamics research. Thus, most of the effort was focused on novel interactive flow field modeling methods based on the manipulation of splines that influence the flow and roughly correspond to streamlines in it. These flows can be superimposed with the well-known source, sink, dipole, and vortex flows. Turbulence is modeled by using stochastic distributions of vortices that are advected with the mean flow. This has several advantages, since the turbulence has a small-scale effect and is continuously defined by functions whereas the mean flow can be given on a three-dimensional grid. The use of flow fields allows any geometric model to be advected with the flow. The flow fields can also be used three-dimensional textures. These flow fields realistically model the motion of fluids: the animator can specify the general motion of fluid without worrying about collision detection within the fluid, maintaining constant volume, or the details of the motion. The result is a method of designing and controlling fluid motion for computer animation that is part of a general, efficient, and realistic method of animating fluids.

6.2 Future Work

A main goal of this thesis is to avoid the ad hoc methods of previous work, and every effort was made to make the methods introduced amenable to easy refinement and extension. It is hoped that the integrated approach of this thesis along with the general hydrodynamic framework invite future work.

The framework was designed to easily integrate flow fields produced by dynamic simulation. Considerable effort went into exploring many rapid and stable numerical solution methods for simple hydrodynamics, but even the simplest hydrodynamic models present formidable challenges. One potentially fruitful avenue of future work: modeling liquid geometrically by oriented particle systems and using the boundary element method to solve the relevant equations. While the boundary element method is currently too slow for computer animation purposes, it may be made more efficient in the near future. Another approach is to use Smoothed Particle Hydrodynamics (SPH) which involves using particles that are filtered point samples. A difficulty with SPH for incompressible free-surface flow is to propose a well-defined problem and derive a stable numerical solution method for it—initial-boundary value problems are notoriously subtle.

Other possible extensions include compressible flow and reactive flow, allowing, for example, fire. Also, closely related to fluid motion is the motion of elastically deformable models. The boundary element method is also applicable to these models. Ultimately, the goal is a unified approach to the modeling of natural phenomena. This thesis is one step on the long path to this goal.

Glossary

advection the flow of a current of fluid or transport by such a flow

Bernoulli equation

boundary layer a region of retarded fluid near the surface of a body which moves through a fluid or past which a fluid moves

compressible flow

continuity equation

Euler equations equations of motion under the assumptions of inviscid flow

Eulerian frame

inviscid zero viscosity

irrotational flow flow without vortices (zero vorticity). More precisely, the curl of the fluid velocity is zero. Intuitively, fluid particles do not spin in irrotational flow.

incompressible flow flow where

Lagrangian frame

laminar flow flow that is steady near boundaries contrasted with turbulent flow

Navier-Stokes equations

potential equation

potential flow

Reynolds number ratio of viscous to inertial forces. Flow becomes turbulent as increases?

steady flow flow where the fluid velocity does not change with time

Stokes derivative

streamline line tangential to fluid velocity. For steady flows, a fluid particle follows a streamline.

turbulent flow

uniform flow flow where the fluid velocity is the same at all points in the fluid

unsteady flow flow where the fluid velocity changes with time

viscosity 2: the property of a fluid or semifluid that enables it to develop and maintain an amount of shearing stress dependent upon the velocity of flow and then to offer continued resistance to flow 3: the ratio of the tangential frictional force per unit area to the velocity gradient perpendicular to the direction of flow of a liquid – called also coefficient of viscosity

vorticity measure of vortical motion; esp: a vector measure of local rotation in a fluid flow

Bibliography

[1] APPEL, D. W., HUBBARD, P. G., LANDWEBER, L., LAURSEN, E. M., MCNOWN, J. S., ROUSE, H., SIAO, T. T., TOCH, A., AND YIH, C. S. *Advanced Mechanics of Fluids.* John Wiley & Sons, New York, 1959.

[2] BARZEL, R., AND BARR, A. H. A Modeling System Based On Dynamic Constraints. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22:4 (August 1988), 179–188.

[3] CRAPPER, G. D. *Introduction to Water Waves.* Ellis Horwood Limited, 1984.

[4] EBERT, D. S., CARLSON, W. E., AND PARENT, R. E. Solid Spaces and Inverse Particle Systems for Controlling the Animation of Gases and Fluids. To appear in *The Visual Computer* (*Expected* November 1993).

[5] FEYNMAN, R. P., LEIGHTON, R. B., AND SANDS, M. *The Feynman Lectures on Physics.* Addison-Wesley, 1963.

[6] FOURNIER, A., AND REEVES, W. T. A Simple Model of Ocean Waves. *Computer Graphics (SIGGRAPH '86 Proceedings)*, 20:4 (August 1986), 75–84.

[7] GOSS, M. E. A Real Time Particle System for Display of Ship Wakes. *IEEE Computer Graphics and Applications*, 10:3 (May 1990), 30–35.

[8] HAHN, J. K. Realistic Animation of Rigid Bodies. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22:4 (August 1988), 299–308.

[9] ISAACS, P. M., AND COHEN, M. F. Controlling Dynamic Simulation with Kinematic Constraints, Behavior Functions and Inverse Dynamics. *Computer Graphics (SIGGRAPH '87 Proceedings)*, 21:4 (July 1987), 215–224.

[10] KASS, M., AND MILLER, G. Rapid, Stable Fluid Dynamics for Computer Graphics. *Computer Graphics (SIGGRAPH '90 Proceedings)*, 24:4 (August 1990), 49–57.

[11] KINSMAN, B. *Wind Waves.* Dover, 1984.

[12] LAMB, H. *Hydrodynamics.* Dover, 1945.

[13] MASTIN, G., WATTERBERG, P., AND MAREDA, J. Fourier Synthesis of Ocean Scenes. *IEEE Computer Graphics and Applications*, 7:3 (March 1987), 16–23.

[14] MAX, N. Vectorized Procedural Models for Natural Terrain: Waves and Islands in the Sunset. *Computer Graphics (SIGGRAPH '81 Proceedings)*, 15:3 (August 1981), 317–324.

[15] MILLER, G., AND PEARCE, A. Globular Dynamics: A Connected Particle System for Animating Viscous Fluids. *Computers and Graphics*, 13:3 (1989), 305–309.

[16] Newman, J. N. *Marine Hydrodynamics.* MIT Press, 1977.

[17] O'Neill, M. E., and Chorlton, F. *Ideal and Incompressible Fluid Dynamics.* Ellis Horwood Limited, 1986.

[18] Peachey, D. Modeling Waves and Surf. *Computer Graphics (SIGGRAPH '86 Proceedings)*, 20:4 (August 1986), 65–74.

[19] Perlin, K. An Image Synthesizer. *Computer Graphics (SIGGRAPH '85 Proceedings)*, 19:3 (July 1985), 287–296.

[20] Platt, J., and Barr, A. Constraint Methods for Flexible Models. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22:4 (August 1988), 279–288.

[21] Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. *Numerical Recipes in C.* Cambridge University Press, 1988.

[22] Reeves, W. T. Particle systems – A technique for modeling a class of fuzzy objects. *Computer Graphics (SIGGRAPH '83 Proceedings)*, 17:3 (July 1983), 359–376.

[23] Shinya, M., and Fournier, A. Stochastic Motion — Motion under the Influence of Wind. To appear in *Proceedings of Eurographics '92* (1992).

[24] Silicon Graphics, I. *Graphics Library Programming Guide.* Publisher, Address, 1991.

[25] Sims, K. Particle Animation and Rendering Using Data Parallel Computation. *Computer Graphics (SIGGRAPH '90 Proceedings)*, 24:4 (August 1990), 405–413.

[26] Stam, J., and Fiume, E. Turbulent wind fields for gaseous phenomena. *Computer Graphics (SIGGRAPH '93 Proceedings)* (August 1993), 369–376.

[27] Stoker, J. J. *Water Waves.* Interscience Publishers, 1957.

[28] Terzopoulos, D., Platt, J., and Fleischer, K. Heating and Melting Deformable Models (From Goop to Glop). *Proceedings of Graphics Interface 1989* (June 1989), 219–226.

[29] Tonnesen, D. Modeling Liquids and Solids Using Thermal Particles. *Proceedings of Graphics Interface 1991* (June 1991), 255–262.

[30] Ts'o, P. Y., and Barsky, B. A. Modeling and Rendering Waves: Wave-Tracing Using Beta-Splines and Reflective and Refractive Texture Mapping. *ACM Transactions on Graphics*, 6:3 (July 1987), 191–214.

[31] Witkin, A., and Kass, M. Spacetime Constraints. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22:4 (August 1988), 159–168.

[32] Wyvill, G., Pearce, A., and Wyvill, B. The Representation of Water. *Proceedings of Graphics Interface '86* (May 1986), 217–222.