Virtual Smoke: An Interactive 3D Flow Visualization Technique

Kwan-Liu Ma

Department of Computer Science
University of Utah
Salt Lake City, Utah 84112

Philip J. Smith

Department of Chemical Engineering
University of Utah
Salt Lake City, Utah 84112

Abstract

This paper introduces a new technique for computer visualization of simultaneous three-dimensional vector and scalar fields such as velocity and temperature in reacting fluid flow fields. The technique, which we call Virtual Smoke, simulates the use of colored smoke for experimental gaseous fluid flow visualization. However, it is noninvasive and can animate, in particular, the dynamic behaviors of steady-state or instantaneous flow fields obtained from numerical simulations. Virtual Smoke is based on Volume Seeds and Volume Seedlings, which are direct volume visualization methods previously developed for highly interactive scalar volume data exploration. We use data from combustion simulations to demonstrate the effectiveness of Virtual Smoke.

1 Introduction

Flow visualization has played an important role in the understanding of fluid mechanical problems. To make the motion of transparent fluid visible, in experimental fluid mechanics, foreign materials are added to the flowing gaseous or liquid fluid. Quantitative data are often obtained by introducing some intrusive measuring instrument, such as a pressure transducer or temperature probe. The physical presence of these foreign materials and measurement devices in the fluid flow may significantly affect the experimental results [12].

While the cost of performing experiments has been steadily increasing, computational costs have been decreasing. Software tools for simulating physical processes have become more accurate, easier to use, and more readily available. These trends have made computational fluid dynamics more attractive for design and problem-solving applications. Large-scale numerical three-dimensional flow simulations can generate large data sets, ranging from several megabytes to even gigabytes in size. To interpret and analyze such data, many computer graphics techniques have been developed. However, most of these techniques fail to incorporate both scalar and vector fields into a single visualization that could reveal the flow features hidden in the large quantity of three-dimensional data.

In this paper, we describe a new technique for visualizing numerical three-dimensional, scalar and vector data. This technique, herein called Virtual Smoke, simulates the use of colored smoke for gaseous fluid flow or dye for liquid fluid flow in experimental flow visualization. For example, it can animate the dynamic behaviors of steady-state flow fields obtained from numerical simulations, or it can show hidden flow characteristics in a single time slice of dynamic flow field calculations. That is, the user is able to interactively select a position in the three-dimensional simulation domain and see the flow and scalar fields developing from that position. On a Silicon Graphics GTX/240, scenes can be updated at a sufficiently satisfactory rate to achieve motion for understanding flow dynamics. Virtual Smoke has been implemented on top of an interactive volume exploration system based on Volume Seeds [8] and Volume Seedlings [3], which are feature extraction techniques previously developed for highly interactive scalar volume data exploration. The ability of simultaneously incorporating scalar and vector field data makes Virtual Smoke especially powerful for but not limited to the visualization of fluid flow fields. For examples, electromagnetic and biofluid fields are other possible applications for Virtual Smoke. In this paper, we show results of the use of Virtual Smoke on data from combustion simulations to demonstrate its effectiveness.

2 Interactive Volume Visualization

Direct volume rendering, creating an image directly from volume data without constructing intermediate graphics primitives, has been shown to be a very effective method for visualizing scalar volume data [4, 19]. An important part of the volume rendering process is

0-8186-2897-9/92 $03.00 © 1992 IEEE

to assign color and opacity maps. Appropriate color mapping creates more understandable images and enhances features of interest. Opacity mapping allows the user to focus on a portion of the data by making uninteresting portions transparent. However, volume rendering is computationally expensive and the rendering time grows linearly with the size of the data set. As a result, many refined algorithms have been developed which exploit the coherence in the data and the rendering process [7]. Special hardware [6, 22] and parallel algorithms [9, 15] have also been designed to offer near real-time visualization of scalar volume data.

We have designed an interactive volume visualization system which takes advantage of the coherence across all possible images from a given viewpoint to provide interactive rendering rates for high quality images. This system consists of four components: a volume renderer, a data slicer, a color map maker and an opacity map maker, each of which runs as a separate process. Interactivity is achieved by caching rendering information independent of color and opacity mappings. That is, the user is able to interactively modify either the color or opacity mapping and see the corresponding changes in the images as the changes in the mappings are made, providing an interactive loop for volume exploration. Currently this system has been implemented using the Silicon Graphics' IRIS Graphics Library (GL), which is also supported on the IBM RS/6000 workstations. The technique has been applied to a variety of applications [8, 10, 11, 2, 14].

2.1 Volume Seeds

Volume Seeds, based on the ray-casting volume rendering algorithm [7], allows the user to perform location-based data exploration and to visualize internal structures in the data which may be obscured by other imaging algorithms. We have integrated this technique into our volume visualization system. Local areas of interest within the volume can be indicated by the user planting a "seed" in the volume. The seed is used to implicitly define a matte volume. Local rendering parameters can then be modified based on location relative to the seed. An example is to accentuate data near the seed point and attenuate data away from the seed point. The resulting image enhances an area, usually a spherical region, around the seed. By adding a binary decision indicating if the sample is in front or behind an imaginary plane through the seed, virtual cut-aways can also be produced in the same way.

2.2 Volume Seedlings

A single seed highlights a spherical region around the seed point. In many applications, however, the shape of the region of interest within the volume is not strictly spherical, but rather is data dependent. Volume Seedlings, an enhancement of Volume Seeds, is to use the seed point as a base from which to sprout a seedling along paths of "maximum interest", thus highlighting a region of interest. The seedling growth algorithm used in Volume Seedlings is voxel based. A priority queue of voxels is maintained determining voxels within the volume which need to be explored. Initially, the priority queue contains only the user specified seed. At each growth step, the highest priority voxel is extracted from the priority queue and its 26 neighboring voxels are examined. The priority assigned to each voxel within the queue is based on the "degree of interest", which could be material classification, gradient magnitude, position or any other predefined function.

The seedling growth process yields a set of voxels, in priority order, defining a region of interest within the volume. The region of interest is highlighted through the use of an opacity matte as for a single seed. The opacity matte volume is based on a function of the distance to the closest voxel of the seedling. By adding one additional byte to the ray-sample cache to hold the distance to the nearest voxel in the seedling, images can be computed incrementally. As each new voxel of interest is extracted from the priority queue, only rays representing pixels which pass near the new voxel need to be processed. The minimum distance from a sample point to any voxel on the seedling is maintained by updating the distance only when the new voxel on the seedling is closer than any previously processed voxels. Details of the Volume Seeds and Volume Seedlings algorithms can be found in [8, 3]. Volume Seedlings, also integrated into our visualization system, have been demonstrated to be effective in visualizing Magnetic Resonance Angiography data for diagnosing malformations such as anuerisms and blockages in blood vessels [3].

3 Flow Visualization

Graphical flow visualization is essential for scientific researchers to understand the results of fluid dynamics simulations in fields such as aeronautics, transportation design, weather forecasting, oceanography, and combustion engineering. To date, there is still no single flow visualization method that has emerged as being the standard due to the complexity of three-dimensional flow fields. Therefore, people have integrated multiple techniques into one visualization system with a uniform interface [1, 13]. Well known flow visualization methods include arrow field, contouring, particle advection, particle tracing, streamlines, etc. Other recently developed techniques include flow ribbons, stream polygon [16], vortex tube [17], spot noise [20], and critical point analysis [5]. A good example using a combination of flow visualization techniques is the visualization of thunderstorm simulation done by NCSA [21].

Direct volume rendering has also been used to visualize scalar flow fields [2, 10, 18]. Carefully selecting color and opacity mapping values can make very effective images. Plate 1 shows a sequence of ray-casting volume rendered images of numerical results from a three-dimensional unsteady compressible flow simulation. For unsteady flow, properties at each point in the flow field change with time. On the other hand, for steady flow, or for unsteady flows at one time slice, properties at each point in a flow field do not change. The three-dimensional dynamic behavior of the flow is difficult to uncover with a single volume rendered image. Instead, a visualization method like particle tracing is usually used. Figure 1 describes these two types of flow calculations and shows where Virtual Smoke can be applied.

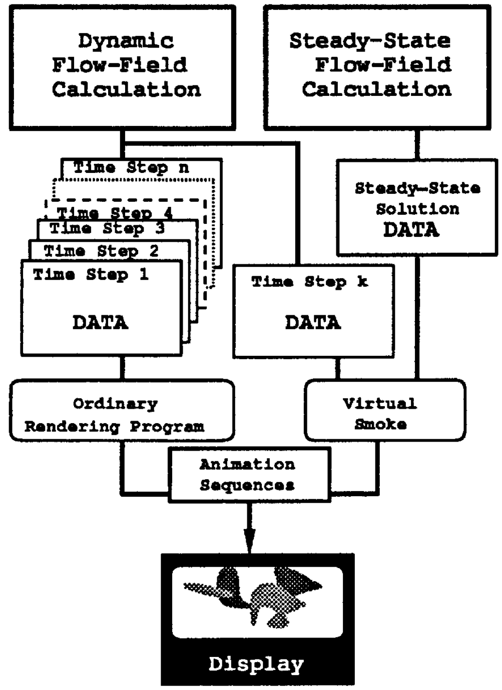

Figure 1: Unsteady (Dynamic) and Steady Flow Calculations and Visualization.

4 Virtual Smoke

Virtual Smoke is a new technique particularly designed for visualizing the dynamic features of three-dimensional vector fields. In the concept of Virtual Smoke, the vector field is visualized through the use of motion. The flow velocity is a vector, and in measuring it we must provide for determining both the magnitude and the direction of the velocity as a function of spatial position and time.

A prototype system has been implemented over the interactive visualization system described in the previous section. Therefore, Virtual Smoke inherits all the features of Volume Seeds and Volume Seedlings. The use of Virtual Smoke begins with the selection of a seed point using the data slicer which is equipped with a combination of three visualization methods: two-dimensional color contour slicing and arrow-field plotting, and isosurface generation. The user can interactively slice through the data volume and pick a point on the current slice plane or isosurface. The seed position is rounded to the nearest voxel position and is passed to the volume renderer. Then the volume visualization process starts by first calculating an initial control volume surrounding the seed point. The control volume grows voxel by voxel based on both the direction and magnitude of the velocity values near the seed point. The display is updated accordingly to show the growth of the seed, which in turn gives a visualization of the vector field, i.e. flow pattern, around the seed point. The scalar field, such as gas temperature or the magnitude of velocity, is mapped to color in order to visualize the vector and scalar field simultaneously.

4.1 Calculation of Control Volume

The rendered control volume is a deformable volume enclosed by a bounding control surface and growing with time in the fixed three-dimensional vector field. Conceptually, "smoke" that is neutral buoyant and is color-coded by the local scalar value is injected into the control volume from the seed point. The control volume contains voxels that would be reached by the injected "smoke" from the seed point after varying time increments if the vector field were fixed. That is, while time proceeds, the control volume grows by tracing where all control surfaces move for a fixed vector field. The voxels in the control volume are ordered according to how long the control surface took to move from the seed point to the location of that voxel, i.e.

```
while (Q is not empty) {
  P = get_event(Q);
  for (each of the 26 neighbors P_i of P) {
    calculate V_i the velocity normal to (P_i - P);
    calculate the propagation time t[P_i] = δ_i/V_i + T[P];
    if (P_i is not in Q)
      insert P_i with t[P_i] into Q;
    else
      if (T[P_i] > t[P_i]) T[P_i] = t[P_i];
  }
  rendering(P);
}
```

Figure 2: The Calculation of Control Volume.

the propagation time. Therefore, instead of using a priority queue as in Volume Seedlings, an event queue Q is used and voxels in queue are ordered based on the propagation time. At the beginning, the seed point S with propagation time equal to zero is inserted into Q and S is the only element in Q. Then the pseudo iterative routine shown in Figure 2 performs the calculation of the control volume and invokes a corresponding rendering routine at the end of each iteration. *Get_event* returns a voxel, P, with shortest propagation time. $T[P]$ is the propagation time needed for the "smoke" to move from the seed point to P; δ_i is the distance between P and P_i; and V_i is calculated as follows:

$$\vec{V}_i = (\vec{V}_{avg} \cdot \vec{N}_i) \times \vec{N}_i$$

where

$$\vec{N}_i = \frac{P_i - P}{|P_i - P|} \quad \text{and} \quad \vec{V}_{avg} = \frac{\vec{V}[P] + \vec{V}[P_i]}{2}$$

Figure 3 shows a slice of a hypothetic control volume where the voxel marked in black and labeled P is the initial seed point. As indicated in the last statement in the *for* loop, if a voxel has been previously visited from a different path, its new propagation time is compared to the previously calculated one; if the new propagation time is shorter, it replaces the old one. This assures correct temporal ordering of the voxels to be rendered.

4.2 Rendering

During each iteration, we select the voxel with the shortest propagation time from the first seed, visit its 26 neighbors, expand the control volume and render a region surrounding that voxel, a procedure similar to Volume Seedlings. Again, the rendering corresponds to casting rays from a rectangular region in

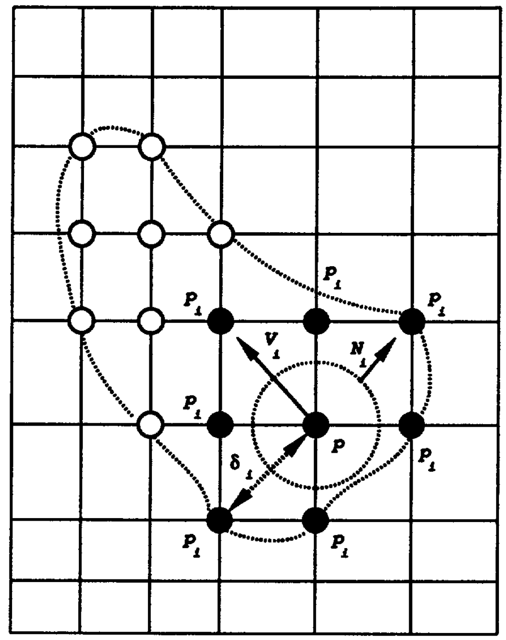

Figure 3: A Slice of a Hypothetic Control Volume.

the image space because only those rays pass near the new voxel need to be processed. As in Volume Seeds and Seedlings, the region near the given voxel position is highlighted according to some parameters interactively controlled by the user. Figure 4 shows a slice through the opacity matte as a function of the distance from a seed. Due to the matte function we use, this region is a cloud like sphere.

By using the ray-sample caching technique and limiting the growth of the control volume at each iteration to 26 nearest voxels around the seed point considered, near real-time animation of the flow as small spherical clouds can be produced. A video rate of ten or more frames per second is achieved on a Silicon Graphics GTX/240. While this speed is adequate for interactive viewing of local flow patterns, a complete voxel-by-voxel growing sequence would take tens of minutes for a large data set. However, a complete growing is usually unnecessary since the most frequent application is the viewing of flow patterns in isolated areas.

4.3 Particle Tracing

Instead of visiting all 26 neighbors, visiting one voxel at a time according to the local velocity can form a trace of particles. The trace is then connected with small color-coded clouds. Ribbon-like paths can also be emulated by using multiple seeds. This capability

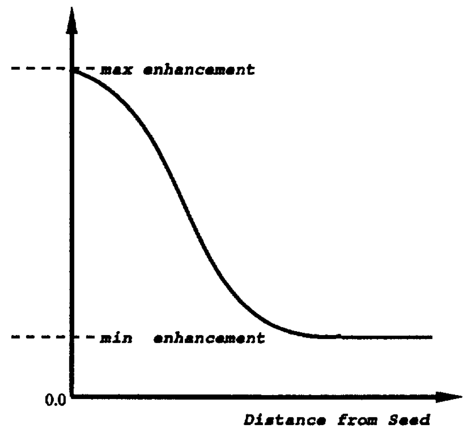

Figure 4: A Graphical Illustration of the Opacity Matte as a Function of the Distance from a Seed.

is also useful to observe complex flow fields as shown in the following section.

4.4 Display of Flow Speed

Incorporating flow speed into the animation is a difficult and computationally expensive task. Either an exhaustive search and rendering algorithm or some kind of delay mechanism in the rendering is needed to show the magnitude of the vector realisticly. First, one could search all voxels that are reached by the injected "smoke" within a user defined and preselected time interval and render only those voxels at each animation update; however this algorithm is nontrivial and expensive. Moreover, for a simulation domain having dramatically different vector magnitudes in different subdomains, the animation could be either too fast or too slow to be useful. Secondly, one could update each animation rendering at a time interval that was proportional to the propagation time and thus proportional to the vector magnitude; however, in order to maintain a near real-time animation of the flow, we cannot afford to intervene any delay into the visualization process with current workstation computational power.

Although we manage accurately the order of the "smoke" movement and thus the direction of the vector field, the magnitude (i.e. flow speed) is not always correctly displayed. If the vector magnitude is used as the scalar field, the colors of the "smoke" directly show its corresponding flow speed. However, this restriction prohibits us from viewing a combination of a vector field with other scalar fields like temperature, pressure or a chemical species concentration. While investigating a solution that is more computationally feasible, we currently use a time display showing how fast or slow the flow simulation time elapses to give the user a feeling of the flow speed, which has been an effective alternative so far.

4.5 Test Results

Three-dimensional computational fluid dynamics applications whether, transient or steady-state, provide velocity vector fields that are often difficult to conceptualize. At each time step in a transient computation the flow field often changes dramatically. Visualizing the direction and magnitude of the velocity field and simultaneously visualizing the effect of the flow on some scalar field at any one time step is the objective of Virtual Smoke.

The samples selected in this section come from the field of combustion engineering. The application is that of a steady-state simulation of combustion performance in three different furnaces. In these combustion chambers the fluid flow affects many other processes including heat transfer and chemical reaction. This computational problem involves discretization of the partial differential equation set using a non-uniformly spaced three-dimensional Cartesian and cylindrical mesh. About 10^6 nodes are needed to resolve important features of the combustion process. Around 60 variables (representing, e.g., components of velocity of the gas, gas pressure, and concentration of various chemical species) are tracked at each node. The trajectories of the particles are followed by introducing them into the flow-field induced by the mixing and reaction process. Their position in turn influences the velocity and pressure fields of the combustion process. The overall strategy is to calculate properties of the gas and use these values to move the coal dust particles. Mathematically the system may be described as a coupled system of nonsymmetric elliptic particle differential equations with algebraic constraints. The ability of visualizing the effect of the flow field on variables such as temperature, pollutant concentrations, radiant energy, etc. is important.

The first example, that of Plate 2, shows the visualization of a simulation of natural gas combustion in a pilot scale furnace with one burner fired from left to right. The computational grid consists of $55 \times 45 \times 45$ data points. The four images show an animation sequence of the flow using Virtual Smoke and by placing a seed point near the burner. In this case, the flow is dominated by a strong central jet that tries to escape through a small exit portal on the right side of the furnace. As a result, the flow field has a large recirculation zone surrounding the central jet which is filled mostly from the gas near the exit. Some significant contributions from the high shear layer near the jet source are also visible. The Virtual Smoke animation allows for visualization of the direction of the velocity, while the colors of the animation show the magnitude of the velocity.

One of the more significant physical questions for which an answer is being sought with the combustion simulation is how to minimize the formation of pollutants such as NO. This can only be achieved by controlling the mixing rates of the burner gases with the surrounding hot gas. The physical objective is to accomplish complete mixing while keeping the temperature as low as possible. Using Virtual Smoke, we can simultaneously see mixing rates and local temperature values. We can look at the mixing of each burner independent of the others and can then decide how to tune the design to minimize pollution formation.

The second example is the visualization of a full-scale simulation of combustion in a utility boiler fired with twelve burners on opposite walls. The computational grid contains $52\times34\times84$ data points. Plate 3 shows, from left to right, the wall structure of the furnace, a combination of isosurface and color-contour slice, and a volume rendered image of the velocity magnitude. In Plate 4, we select three key frames from an animation sequence from the Virtual Smoke visualization displaying flow motion with color mapping of gas temperature. The seed point was placed near one of the burners as depicted in the left most images. In this case the flow patterns are very complex. the boiler being simulated is more than 100 meters tall and the multiple burners result in a flow field that is difficult to interpret. The Virtual Smoke technique allows the user to interactively place seeds at various starting points to interpret the complexities of the flow and scalar fields. Even from this limited number of views from the sequence it can be seen how the flow from one burner is swept mostly upwards and out the exit but how some bleeds into the recirculation zone in the hopper at the bottom of the furnace.

The last example is the simulation of a cyclonic combustion furnace. This furnace, although smaller than the previous two (being about one meter long), is designed to have a circuitous flow path. The flow enters on the side of the cylindrical cyclone and around the perimeter then back up the center draft tube. The computational grid consists of $86\times37\times37$ data points. In Plate 5, the upper image depicts the wall structure and the lower image shows color-mapped velocity magnitude of this furnace using volume rendering. We use this data set to show the capability of the Virtual Smoke algorithm to perform particle tracing. The four images in Plate 6 shows traces of particles seeded at different locations.

5 Conclusions

We have presented Virtual Smoke, a new technique for visualizing vector and scalar data simultaneously. The most powerful feature of this technique is the visualization of the vector field through computer animation. Virtual Smoke can also emulate particle tracing without using geometrical primitives. We have incorporated Virtual Smoke in a visualization system that was previously implemented for interactive exploration of scalar volume data on a graphics workstation. Applying this new visualization technique to steady flow data from combustion simulations has demonstrated its effectiveness.

The idea of Virtual Smoke has evolved from Volume Seeds and Volume Seedlings; however, its implementation is not tied to them. The current implementation has not only demonstrated the concept but has also shown the usefulness of Virtual Smoke for visualizing complex vector and scalar fields. We are currently investigating the implementation of Virtual Smoke using other volume rendering methods, incorporation of topology analysis calculations to assist the user in locating seed positions, and applying it to a wider range of vector data from other disciplines.

Acknowledgments

This work was supported in part by NSF/ACERC and an IBM grant for scientific visualization. Thanks go to Michael Cohen and Jamie Painter for their guidances during the development of Volume Seeds and Volume Seedlings. Mihir Mehta integrated an isosurface program into the data slicer that was originally implemented by Elena Driskill. Video production assistance was provided by Robert McDermott and James Rose.

References

[1] Bancroft, G., Merritt, F., Plessel, T., Kelaita, P., McCabe, R., and Globus, A. FAST: A Multi-Processed Environment for Visualization of Computational Fluid Dynamics. In *29th Aerospace Sciences Meeting* (January 1991), American Institute of Aeronautics and Astronautics. AIAA Paper 91-0793.

[2] Clarkean, R., and Ma, K.-L. Direct Numerical Simulation and Visualization of a 3D Planar Mixing Layer. In *Proceedings of 1991 AIAA Fluid*

Dynamics Conference (1991), American Institute of Aeronautics and Astronautics.

[3] COHEN, M. F., PAINTER, J. S., MEHTA, M., AND MA, K.-L. Volume Seedlings. In *Proceedings of ACM Symposium on Interactive 3D Graphics* (March 1992).

[4] DREBIN, R. A., CARPENTER, L., AND HANRAHAN, P. Volume Rendering. *Computer Graphics (Proceedings of SIGGRAPH 1988) 22*, 4 (August 1988), 65–74.

[5] HELMAN, J. L., AND HESSELINK, L. Visualization of Vector Field Topology in Fluid Flows. *IEEE Computer Grahics and Applications 11*, 3 (May 1991), 36–46.

[6] KAUFMAN, A., AND BAKALASH, R. Memory and Processing Architecture for 3D Voxel-Based Imagery. *IEEE Computer Graphics and Applications 8*, 6 (November 1988), 10–23.

[7] LEVOY, M. Efficient Ray Tracing of Volume Data. *ACM Transactions on Graphics 9*, 3 (July 1990), 245–261.

[8] MA, K.-L., COHEN, M., AND PAINTER, J. Volume Seeds: A Volume Exploration Technique. *The Journal of Visualization and Computer Animation 2*, 4 (1991), 135–140.

[9] MA, K.-L., AND PAINTER, J. Parallel Volume Visualization on Workstations. Computers and Graphics (under review), 1992.

[10] MA, K.-L., AND SIKORSKI, K. A Distributed Solution and Visualization for 3D Flow Simulation. In *Proceedings of the 5th SIAM Conference on Parallel Processing for Scientific computing* (1992), the Society for Industrial and Applied Mathematics.

[11] MA, K.-L., SMITH, P., ADAMS, B., AND SIKORSKI, K. Distributed Combustion Simulation. Parallel CFD'92 (under review), 1992.

[12] MERZKIRCH, W. *Flow Visualization*, 2 ed. Academic Press, Inc., 1987.

[13] MILLER, S. L., AND EDWARDS, D. E. VISA: An Interactive Graphics System for Scientific Visualization. In *29th Aerospace Sciences Meeting* (January 1991), American Institute of Aeronautics and Astronautics. AIAA Paper 91-0795.

[14] MORIN, P. J., AND TANIMOTO, T. Visualizing the Three-Dimensional Structure of the Earth's Upper Mantle. Tech. rep., Army High Performance Computing Research Center, 1991. Preprint 91-118.

[15] SCHRÖDER, P., AND B., S. J. Fast Rotation of Volume Data on Data Parallel Architectures. In *Proceedings of Visualization'91* (October 1991), pp. 50–57.

[16] SCHROEDER, W., VOLPE, C., AND LORENSEN, W. The Stream Polygon: A Technique for 3D Vector Field Visualization. In *Proceedings Visualization'91* (1991), G. M. Nielson and L. Rosenblum, Eds., IEEE Computer Society Press, pp. 126–132.

[17] SILVER, D., GAO, M., AND ZABUSKY, N. Visualizing Causal Effects in 4D Space-Time Vector Fields. In *Proceedings Visualization'91* (1991), G. M. Nielson and L. Rosenblum, Eds., IEEE Computer Society Press, pp. 12–16.

[18] SWANN, P., AND SEMWAL, S. K. Volume Rendering of Flow-Visualization Point Data. In *Proceedings Visualization'91* (1991), G. M. Nielson and L. Rosenblum, Eds., IEEE Computer Society Press, pp. 25–32.

[19] UPSON, C., AND M., K. V-buffer: Visible Volume Rendering. *Computer Graphics (Proceedings of SIGGRAPH 1988) 22*, 4 (August 1988), 59–64.

[20] VAN WIJK, J. J. Spt Noise - Texture Synthesis for Data Visualization. *Computer Graphics 25*, 4 (July 1991), 309–318.

[21] WILHELMSON, R., JEWETT, B. F., SHAW, C., WICKER, L. J., ARROTT, M., BUSHELL, C. B. BAJUK, M., THINGVOLD, J., AND YOST, J. A Study of te Evolution of a Numerically Modeled Severe Storm. *The International Journal of Supercomputer Applications 4*, 2 (Summer 1990), 20–36.

[22] YOO, T., NEUMANN, U., FUCHS, H., PIZER, S., CULLIP, T., RHOADES, J., AND WHITAKER, R. Achieving Direct Volume Visualization with Interactive Semantic Region Selection. In *Proceedings of Visualization'91* (October 1991), pp. 58–68.

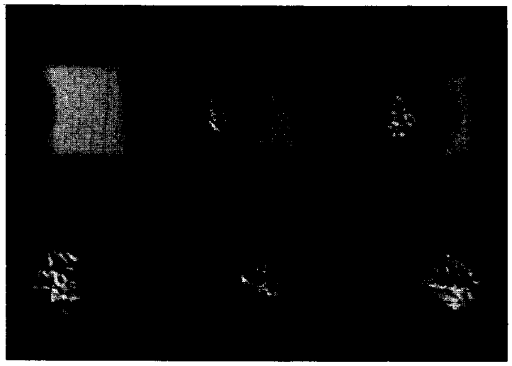

Plate 1: Animation of 3-D unsteady flow.

Plate 2: Animation of a pilot furnace simulation.

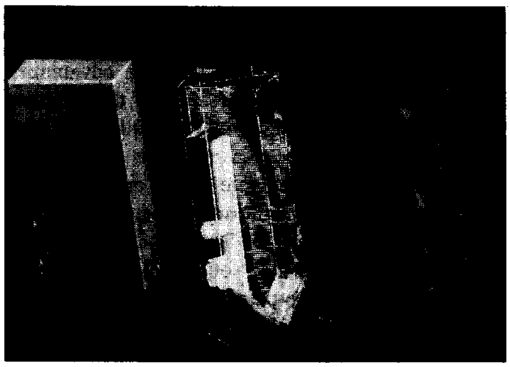

Plate 3: Visualization of a full-scale furnace simulation.

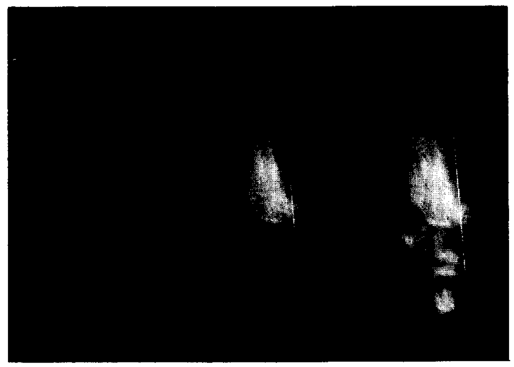

Plate 4: Animation of a full-scale furnace simulation.

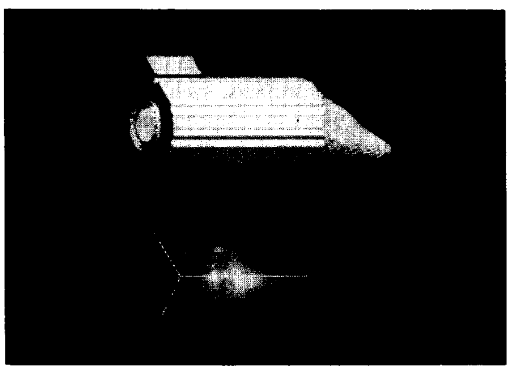

Plate 5: Visualization of a cyclonic combustion furnace.

Plate 6: Particle tracing using virtual smoke.

(See color plates, p. CP-6.)