The Virtual Windtunnel: An Environment for the Exploration of Three-Dimensional Unsteady Flows

Steve Bryson† and Creon Levit††

RNR Technical Report RNR-92-013, October 1991

Applied Research Branch, Numerical Aerodynamics Simulation Division
NASA Ames Research Center
MS T27-A
Moffett Field, Ca. 94035
bryson@nas.nasa.gov

Abstract

We describe a recently completed implementation of a virtual environment for exploring numerically-generated three-dimensional unsteady flowfields. A boom-mounted six degree of freedom head-position-sensitive stereo CRT system is used for viewing. A hand position sensitive glove controller is used for injecting various tracers (e.g.. "smoke") into the virtual flowfield. A mutiprocessor graphics workstation is used for computation and rendering. We describe our techniques for visualizing unsteady flows and discuss the computer requirements for a variety of visualization techniques. These techniques generalize to visualization of other three-dimensional vector fields.

---

† Employee of Computer Sciences Corporation. Work supported under government contract NAS 2-12961
†† Employee of NASA

1: Introduction

Visualization of the three-dimensional flowfields that are the output of numerical flow simulations is difficult. In the velocity vector fields that result from these computations, complicated geometrical and topological situations abound. For example, multiple vortices, recirculation bubbles, and chaotic flows within vortex breakdown have all been observed in computer simulations of *steady* three dimensional fluid flows. The complexity of of three-dimensional unsteady flow patterns is so great that new techniques are required to effectively visualize them. This paper describes an application of virtual environment techniques to visualizing these complex flows.

Unless otherwise noted, when we refer to a *flowfield*, we mean a numerical solution to a three-dimensional computational fluid dynamics (CFD) simulation, and in particular, the time-dependent velocity vector field part of the solution. Previous work in the visualization of numerically computed unsteady fluid flows includes [1][2] and [3].

A fruitful area to search for new methods of numerical flow visualization is among the classical physical techniques - those used to visualize real flows in real wind (or water) tunnels [4]. Smoke injection, dye advection, time exposure photographs, and the placement of tufts or streamers into the flow are examples of these classical techniques. Some additional physical flow visualization techniques are Schlieren interferometry, laser sheet illumination, stroboscopic illumination, and injection of tracers sensitive to fluid properties such as temperature. Computational analogues of these techniques are all feasible using modern high performance graphics workstations and/or distributed computing. These computational analogues of classical windtunnel techniques may be useful in visualizing other vector fields as well.

The computer system requirements for unsteady flow visualization in a virtual environment are substantial. They include speed of computation, ability to quickly render high-resolution graphics, and massive data storage and retrieval capabilities. The amount of solution data produced by a single three-dimensional unsteady fluid dynamics calculation can be quite large - several thousand megabytes or more. High performance graphics workstations have now reached the level where real time interactive exploration of some three-dimensional unsteady flowfield solutions is possible.

Virtual environments are a new approach to user interfaces in computer software. This approach involves integrating a variety of input and display devices to give the user the illusion of being immersed in an interactive computer generated environment. The computer generated scene is displayed in stereo to create the illusion of depth, and the scene is rendered from a point of view that tracks the user's head. The user also has an input device, typically an instrumented glove, through which she can have the experience of directly manipulating objects in the computer generated environment.

We feel these techniques are useful in visualizing complex fluid flows using computer graphics. The stereo head-tracked display is a very effective way of displaying three-dimensional information. Input via the glove is a useful and intuitive way to position and reposition the various flow visualization tools. The idea is to create the illusion that the user is actually in the flow manipulating the visualization tools (figure 1). Unlike someone in a real flow field, however, the user's presence in no way disturbs the flow. Thus, sensitive areas of flow, such as boundary layers and chaotic regions, can be investigated easily. Further, since the flow is precomputed, it can be investigated at any length scale, and with control over time. The time evolution of the flow can be sped up, slowed down, run backwards, or stopped completely for detailed examination.

Figure 1: The Virtual Windtunnel in use, with the flow around the space shuttle.

The flowfields considered in this paper are pre-computed solutions of the time-accurate Navier-Stokes equations of fluid motion. These unsteady flowfields are represented as a collection of successive three-dimensional velocity vector fields. Each of these velocity vector fields is considered as a timestep.

We describe the visualization structures and their interfaces in our environment. We then describe the implementation - first the hardware and then the software. We review the performance of our implementation, we conclude with some comments on what we have learned and, finally, we discuss our future plans.

2: Visualization tools and interfaces

2.1 Definitions

As described in section 1, the tools considered in this paper for visualizing unsteady velocity vector fields are inspired by classical techniques used in real wind and water tunnels. Currently, the visualization methods, or tools, that we have implemented are tufts, streaklines, particle paths, streamlines, and material tracking.

The simplest visualization technique we use is the placing *tufts*, or small vanes, into the flowfield [5]. These tufts, each anchored at a particular location in space, allow direct visualization of the changing velocity vector at that location.

A *streakline* is formally defined as the locus of infinitesimal fluid elements that have previously passed through a given fixed point in space [6]. Informally, a streakline is the evolving curve you obtain if you continuously inject a stream of tracer particles into the flow from a fixed location. In water tunnels, streaklines are usually visualized by generating hydrogen bubbles rapidly, one after another, at a single place by means of an electrolytic wire. Thus, we choose to call the seed point for a streakline a *bubbler*.

A *particle path* is formally defined as the locus of points occupied over time by a given single, infinitesimal fluid element [6]. Informally, a particle path is the curve you obtain if you take a "time exposure photograph" of the motion of a single small particle injected into the flow.

A *streamline* is formally defined as the integral curve of the instantaneous velocity vector field that passes through a given point in space at a given time [6]. Despite their transitory and global nature, streamlines can be visualized in water tunnels by having many tracer particles in the flow, and taking a brief time exposure photograph. The many short, straight particle paths so obtained can be connected to form streamlines.

Material tracking, or dye blob advection, consists physically of following the evolution of an initially specified finite volume of fluid. Numerically, it consists of calculating and displaying the successive positions of a large number of particles ("a blob of dye") that have some initially specified, usually rather compact, distribution.

For steady flows, streaklines, streamlines, and particle paths all coincide. However, for unsteady flows, they are distinct (see figure 2). This has lead to to some confusing terminology in the numerical flow visualization literature, since most previous work has been done on steady flows.

Figure 2: Streamlines, streaklines and particle paths from the same point at the same time in a two-dimensional flow.

2.2 Implementation

With the exception of tufts, the numerical flow visualization techniques mentioned above involve injecting virtual massless point particles into the flow and integrating their trajectories. The distinction between these techniques is in how the particle positions are integrated relative to the time evolution of the field, and how they are displayed. We call the point of injection for a particular tool the *seed point* for that tool. Streaksurfaces and streamsurfaces are the generalization of streaklines and streamlines, respectively, obtained by specifying a set of seed points arrayed along a line or curve. Particle trajectories may be generalized similarly. The various computations used for each tool will now be outlined:

Streaklines consist of point particles, or *bubbles*, all of whose positions change at every timestep. New bubbles are continually emitted from the seed point, or bubbler, at a specified rate, usually once per timestep. The position of a bubbler itself is considered static.

A bubble is a particle with a position (x,y,z). At each time step, the new position of the bubble is computed by integrating the velocity vector field over the current time step and updating the position of the bubble. This position is displayed at the next time step, during which the bubble is again moved. A bubbler is a tool at a seed point that continually emits bubbles, each of which is treated independently. Typically, a bubbler emits a certain number of bubbles, after which its oldest bubbles are recycled back to the seedpoint in the order they were emitted. For our purposes, a streakline, at any timestep, is the current positions of all bubbles emitted by a particular bubbler (figure 3).

A dye blob injection is a large number of bubbles that are placed into the flow at a particular time step. Each bubble is placed at a location determined by the shape of theinitial dye blob, which may be one, two, or three dimensional in extent. The initial locations of the bubbles in a dye blob are all different. After injection, the bubbles in the blob are moved in exactly the same way as the bubbles in a streakline, except that no new bubbles are emitted.

A particle path is akin to a time exposure of the trajectory of a particle that is released at a particular seed point at a particular time. In an unsteady flow the particle path from a particular seed point will vary depending on

Figure 3: Streaklines of the flow around the tapered cylinder rendered as smoke

when the particle is emitted. The particle path is computed by integrating the position of the seed point through the unsteady velocity field over each successive time step, and storing its successive positions into an array. The trajectory is not displayed until the the integration is complete, at which time the trajectory is displayed in full. The complete path of a particle emitted at the current timestep is computed and displayed anew at each timestep. It is displayed as a curve extending from the seed point, representing the path that the particle currently at the seed point will take.

A streamline is computed like a particle path, except that the integration is carried to completion using only the single vector field corresponding to the current time step. The streamline is non-physical in the sense that no fluid element actually follows its trajectory. Nevertheless, streamlines, being integral curves, do give insight into the global structure of the velocity vector field (figure 4).

Tufts are simply very short streamlines. They are usefully approximated by drawing the velocity vectors themselves.

Figure 4: Streamlines of the flow around the tapered cylinder at two successive moments of time.

2.3 Interface

The use of these tools primarily involves the placement of their seed points and the viewing of the graphic objects that result. There are several approaches to the placement of seed points. One approach is to have the computer automatically place them based on analysis of the flow field. For example, seed points may be placed near critical points of the vector field topology [7], or near local maxima of interesting scalars such as helicity [8]. A second approach is to have the user specify, in advance, the positions of the seedpoints, usually as a textual arguments to some command. Though this may sound primitive, it is often most useful. A third approach is simply to give the user flexible, rapid, interactive control over the placement of seedpoints. Our virtual environment supports these last two approaches.

Besides rapid placement of seed points, our virtual environment allows for quick, intuitive repositioning and deletion of existing seed points. Multiple seed points can be grouped together into *rakes*, and repositioning and deletion of these rakes is supported. This is important because the visualization of the flow may involve large numbers of each of these tools. For example, our experience is that about 40 bubblers emitting 100 bubbles each is a minimum requirement for effectively visualizing interesting features of a flow using streaksurfaces. Further, moving these bubblers to another location may reveal previously unseen flow features. Thus, the ability to move rakes of bubblers is an important capability. These considerations apply to particle paths, streamlines, and tufts as well.

Various properties of the tools, such as the number of bubblers in a rake or the length of the streamlines, are controlled via a screen/text interface outside the virtual environment. Using the boom display system (see section 4), we find it fairly convenient to enter and exit the virtual environment to control various aspects of the visualization tools.

Since fluid dynamic phenomena occur over a large range of scales (several orders of magnitude in space) navigation through the virtual environment takes on new difficulties not encountered in, say, architectural walkthroughs. Thus, in additional to “standard” head-position and head-orientation sensitive viewing capabilities, our virtual environment supports the ability to rapidly change one's scale, or the scale of the environment. Thus, a user of the environment can shrink herself so that she is completely surrounded by some small vortex, or enlarge herself so that the entire flowfield fits within her hand.

The flow visualization tools described in section 2.2 produce three dimensional structures that may wind through space in complex and even chaotic ways. To get a good mental picture of the flowfield, cues illuminating the three dimensional geometry are important. Without these cues, ambiguity can result, leading to poor perception of the flowfield. In our virtual environment we use a combination of several techniques to disambiguate three dimensional structures. Real-time three dimensional rendering in response to head position and orientation provides perspective, motion parallax, and vestibular cues. Z-buffering enhances the realism of the image through selective obscuration. Wide field-of-view optics provide input to the peripheral visual field and give a realistic, compelling optical flow. Stereo display provides binocular parallax and further widens the field of view. Finally, feedback as to the position of the hand and animation of the streamlines and particle paths themselves provide additional cues.

3: Hardware

Perhaps the most interesting hardware component of our virtual environments the boom-mounted display (see figure 5). This boom supports two small CRTs on a counterweighted yoke attached through six joints to a base. It is manufactured by Fake Space Labs of Menlo Park CA., and fashioned after the prototype developed earlier by Sterling Software, Inc. at the VIEW lab at NASA Ames Research Center [9].

Figure 5: Boom and glove hardware interface to the Virtual Windtunnel

The boom is an alternative to the popular head-mounted LCD display systems that were pioneered at the VIEW lab [10] and are now widely used. The main advantage of the boom is that real CRTs can be used for display in spite of their mass, since none of the weight of the displays is born by the user. CRTs have much better brightness, contrast, and resolution than standard liquid crystal displays.

The CRTs are mounted on the “head” of the boom, along with the wide field optics and a multi-function handle. Six degrees of freedom of motion are provided by the the gimbals and joints of the boom, in a smooth and force-free manner. Within a very wide range, the user can continuously change the three dimensional position and orientation of the head of the boom. The position and orientation information is based on the the current state of the six joints angles. These angles are sensed by optical encoders at the joints and fed into a microprocessor in the base of the boom, which formats the information and sends it out an RS232 port. No magnetic field emitters or sensors are used, and hence the boom information is precise, repeatable, and insensitive to the electromagnetic environment. Calibration is trivial.

Currently, the CRT monitors on the boom are monochrome. The boom accepts two RS170 video signals, one for each eye.

The motion of the boom is relatively effortless and completely smooth. First-time users are universally surprised that a structure this size moves so easily. Admittedly, compared with head-mounted displays, the footprint of the device is large and the freedom of motion restricted. But when used from a sitting position in a wheeled office chair, it provides ample freedom of motion. With the user standing it is quite usable as well. There are no straps, no weight on the head, and it is easy to disengage from the device and hand it to another user.

In addition to the user's head position and orientation, the user's hand position, orientation, and finger joint angles are sensed using a VPL dataglove™ model II, which incorporates a Polhemus 3Space™ tracker. The finger joint angles are combined and interpreted as gestures. The glove requires recalibration for each user, and the polhemous tracker on the glove is, unfortunately, sensitive to the room's electromagnetic environment. Nevertheless, this part of the system works reliably and satisfactorily once calibrated for a users hand and a room's magnetic peculiarities.

The keyboard and mouse are also used as input devices to the virtual environment. The boom can be easily swung away from the user's eyes and her attention refocussed on the normal computer screen. The user, who is seated, can then return to typing and interacting with the computer in the usual way. For small amounts of typing, and for controlling the mouse, the glove need not be removed, since it is quite thin and flexible.

The computational and rendering for our virtual environment is provided by a Silicon Graphics Iris 380 VGX system. This is a multiprocessor system with eight 33 MHz RISC processors (MIPS R3000 CPUs with R3010 floating point chips). The performance of the machine is rated at approximately 200 Million instructions per second (200 VAX MIPS) and 37 million floating point operations per second (37 64-bit linpack MFLOPS). Our system currently has only 48 MBytes of memory.

The VGX has parallel hardware rendering pipelines. The rated graphics performance of our system is around 800,000 small 3D triangles transformed, clipped, projected, lit, shaded, and displayed per second. The system has over 200 bits per pixel of frame buffer memory. We make use of only 48 bits per pixel - two buffers each of eight bits of red and eight bits of blue (double buffering), and 24 bits of Z-buffer.

Stereo display on the boom is handled by rendering the left eye image using only shades of pure red (of which 256 are available) and the right eye image using only shades of pure blue. When the blue (second, right-eye) image is drawn, it is drawn using a “writemask” that protects the bits of the red image. The Z-buffer bit planes are cleared between the drawing of the left- and right-eye images, but the color (red) bit planes are not cleared. Thus, the end result is separately Z-buffered left- and right-eye images, in red and blue respectively, on the screen at the same time with the appropriate mixture of red and blue where the images overlap.

The 1024x1280 pixel RGB video output of the VGX is converted into RS170 component video in real time using a scan converter. While the VGX can put out RS170 directly by setting a software switch, we have found that scan converting the higher resolution 1024x1280 image using dedicated hardware provides spatial and temporal antialiasing, and consequently noticeably better image quality when viewing with the boom. The red RS170 component is fed into the left eye of the boom, and the blue RS170 component into the right eye. The sync is fed to both eyes. Since the boom CRTs are monochrome, we see correctly matched (stereo) images.

The configuration of our system operating in virtual reality mode is drawn in figure 6.

4: Virtual Environment Software Architecture

The graphics are rendered in stereo from a point of view determined by the boom. The interface to the flow visualization tools is based on the glove position and gesture. There is also an interface allowing the flow data to be moved relative to the user.

The position and orientation of the CRTs on the boom head are determined by optical encoders mounted on the six boom joints. The output of each encoder is linearly related to its respective joint angle. These six joint angles are read by the host computer system and are converted into a standard 4x4 position and orientation matrix. This conversion is the result of six successive translations and rotations. The rotations are rotations about the local axis of the corresponding joint by the angle read at that joint, and the translations are by the distances between joints. In the position and orientation matrix, the position is measured in meters from the base of the boom.

The graphics are rendered in stereo from the point of view of the boom's viewers by inverting the boom's position and orientation matrix, translating to the left or right by half the distance between the eyes (depending on which eye's view is being drawn), and multiplying by a precomputed perspective matrix. The resulting matrix is put on the graphics transformation stack before the rendering of graphics for each respective eye. Thus, the entire view must be rendered twice.

The alignment of the resulting images in the viewer must be correct to obtain a proper stereo effect. This is accomplished by defining separate viewports for rendering

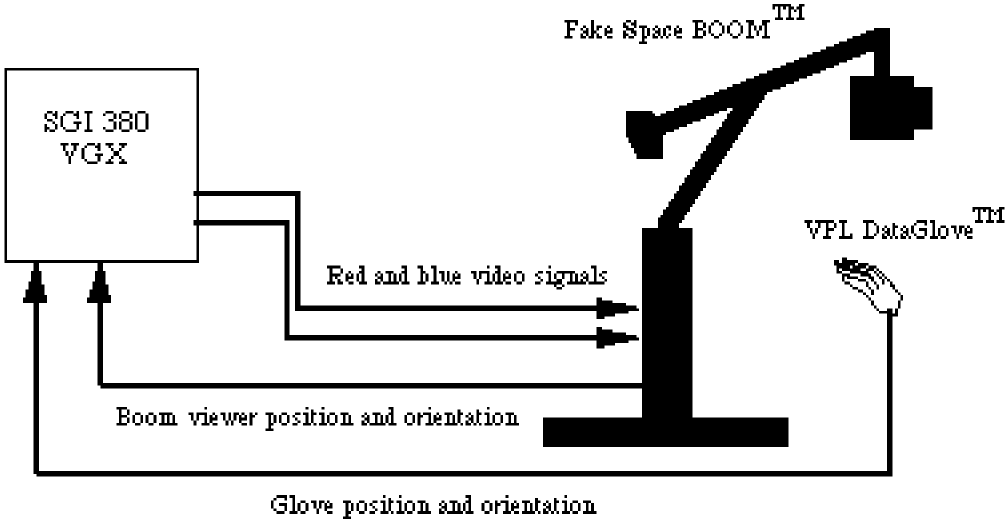

Fig 6: The hardware configuration of the virtual windtunnel system.

the graphics for each eye, with the horizontal position of each viewport controlling the alingment.

Before rendering the graphics data, another transformation embodying a rotation and translation is concatenated onto the transformation stack. This allows the data to be in an arbitrary position and orientation with respect to the coordinate system of the boom. This transformation can be determined in a variety of ways. In particular it enables the manipulation of the entire graphics data via the VPL dataglove. This transformation is called the *data coordinate transformation*.

The primary use of the VPL dataglove is in the placement of rakes of seed points for the various visualization tools. The interface is gesturally based. When the glove position is matched with an existing seed's position and the glove gesture is that of a fist, the seed point is "picked up" and follows the position of the glove until the fist gesture is released. Throughout this time the graphic representation for this seedpoint is recomputed and rendered, allowing the user to observe the tool as it is moved from place to place. When another gesture is performed, a new tool seed point is placed at the current position of the glove. This interface has also been generalized for rakes.

The above interface actually spans three coordinate systems: the glove position is in boom coordinates, the graphics data is in data coordinates, and the streamlines are in computational coordinates. To transform from boom coordinates to data coordinates, the glove position vector is multiplied by the inverse of the data coordinate transformation. The resulting vector is in data (x,y,z) coordinates.

The seed point positions are in computational (ξ,η,ζ) coordinates, which are defined on a discrete curvilinear computational grid. The values (x,y,z) in data coordinates for the grid vertices are stored in a three-dimensional array. The (ξ,η,ζ) values corresponding to the glove's (x,y,z) coordinates are determined using a table search and then refined by inverse interpolation. First the nearest grid vertex to the given (x,y,z) is determined through a simple two step search of the vertex array. This gives a (ξ,η,ζ) value for that vertex. A first order inverse interpolation is then performed using the (x,y,z) values of the neighboring (ξ,η,ζ) points to determine the (ξ,η,ζ) values of the glove position. We are aware that this technique fails on grids that are simultaneously highly curved and stretched, and we are presently implementing a more general method.

Once the seed point's computational space coordinates are known, these coordinates are used as initial conditions for integration by a visualization tool, as described in section 2. Trajectories are generated by integrating a system of three ordinary differential equations:

$$\begin{aligned} d\xi/dt &= u(\xi, \eta, \zeta, t) \\ d\eta/dt &= v(\xi, \eta, \zeta, t) \\ d\zeta/dt &= w(\xi, \eta, \zeta, t) \end{aligned}$$

where u, v, and w, are the components of the velocity vector, in computational coordinates, at the point (ξ,η,ζ) and the time *t*. Values within a grid cell are obtained by trilinear interpolation of the values at the cell vertices.

In all cases, we use second-order accurate Runge-Kutta integration with a fixed step size. An integration is considered complete when either a specified maximum number of steps have been computed, or a position exceeds the boundaries of the grid, or the "end of time" is reached for non-periodic flows. All integration is performed in the computational coordinate system

After integration, the computational coordinates of each visible point are transformed back to physical space, and the points are then displayed, possibly connected by lines. The whole procedure is fast enough to drag rakes of particle paths, streamlines, or streaklines through the field interactively, and it is inaccurate only near extreme velocity or metric discontinuities.

While performing the above computations, the vector field must be in physical memory. Since second order Runge-Kutta is a predictor method, the vector field data for both the current timestep and the next timestep are required. Since we desire each successive timestep be computed and displayed at the rate of at least eight frames per second, the entire vector field data set for all time is kept resident in physical memory. Our disk transfer rates cannot support the alternative.

5: Results

The implementation described in this paper has been used to visualize the three-dimensional unsteady flow past a tapered cylinder computed by Jespersen and Levit [11]. This solution of the compressible Navier-Stokes equations consists of 879 successive time steps on a 64 x 64 x 32 computational grid (131072 grid points per timestep). Each timestep contains about 1.5 megabytes of velocity data. Thus, the total size of the unsteady velocity field is over 1000 megabytes. In order to fit this entire data set into the 48 megabyte memory of our system, it is subsampled by a factor of 2 in each spatial dimension (for a total data reduction by a factor of eight) and then truncated to a 32 x 25 x 4 grid (after subsampling). The truncation is centered around a known interesting region of the flow. This results in a 33.5 megabyte data file.

Numerical integration is distributed over the eight processors in the computer. In the above case, when as many as 4,000 bubbles are present in the flow, the frame rate is about eight frames per second. This performance is obtained entirely in C with standard compiler optimizations. When numerical integration is halted, the frame rate increases.

When we compare the flow phenomena we see in our system to those seen by Jespersen and Levit, it becomes apparent that the spatial subsampling and truncation causes many subtle features to vanish. The large scale features, however, are readily apparent and exploration with the various tools is fruitful.

6: Performance Issues and Future Directions

The study of interesting unsteady flows involves very large data sets. The full tapered cylinder data set considered above is small, as these data go. Other data sets have larger grids, multiple grids, and more time steps. A typical engineering calculation may have two million gridpoints per timestep and thousands of timesteps. Thus, the requirement that the entire data set be resident in physical memory is clearly an impediment to progress.

It may be desirable to trade disk bandwidth for memory. One approach is to store only the solution for the current time step, and replace it by reading new solutions from disk as virtual time advances. Higher order integration methods that require two or more timesteps simultaneously in memory still only require one new solution per timestep.

For this approach to be successful, the system must be able to read one 3D velocity field from disk for each frame of graphics displayed. For "slow-motion" effects, the required data rates are less, since several frames of flow visualization can be generated by interpolating between the solutions at two different timesteps. However, the most common case is where a new timestep worth of data needs to be read in in order to generate a new frame of graphics.

Since the virtual windtunnel uses a head-position-sensitive display, the frame rate must not be allowed to slow down excessively. A good rule of thumb is to require at least ten frames per second. Thus, the disk bandwidth required for applying this "out of core" approach is to the full tapered cylinder dataset is: 1.5 Mbytes/timestep * 10 timesteps/sec = 15 Mbytes/sec.

For this small model problem, a modern superworkstation with striped disks might attain the required performance. For larger problems, it is clear that this approach requires a mini-supercomputer or better.

Thus, it appears that in the future we may want to distribute the virtual windtunnel software between a supercomputer and a graphics workstation, or resort to off-line precomputation of the visualization, and use our specialized hardware largely for viewing the results

The major computational component of the real-time visualization process in the virtual windtunnel is numerical integration of streaklines, streamlines, and particle paths. This numerical integration consists largely of evaluating the integrands, which involves trilinear interpolations. Each second-order Runge-Kutta timestep advance for each particle involves about 200 floating point operations (this includes transforming the particle's position back to physical coordinates for rendering). Again, keeping in mind the desired 10 frames per second, and assuming that the 380 VGX workstation is capable of 20 MFLOPS sustained, we can expect to process 10,000 particles per frame. Our current performance is about half of this, and can doubtless be improved.

How many particles are there in a real puff of smoke, or a real blob of dye? A large number of particles aids visualization, especially for bubblers, since the individual particles in a streakline usually spread too far apart to be connected by lines. Previous experience has shown us that 30,000 particles per frame may not be enough. 100,000 per frame is a useful number, but that requires 200 MFLOPS to support ten frames per second. Thus, the integration of particle positions might profitably be distributed to a faster machine as well.

Virtual Environments are a tantalizing new medium that may become important to scientists and engineers. Though they are currently rather lackluster in terms of performance and image quality, the situation in remeniscent of that in scientific visualization say, ten years ago, when only slow, relatively low quality output devices were available. It was not that long ago when scientific visualization on computers was done by clever overprinting with lineprinters, or if one was really lucky, by using five different colored pens on a calcomp plotter. Virtual environments can be viewed as extensions of the trends that have already been in existence for years: Faster 3D rendering, bigger screens, the "desktop" methaphor, and much else. The “sense of presence” produced by even today's virtual environments certainly has implications for entertainment, and possibly for education. How might they effect the way we do science and engineering? That is the question we are continuing to address in the development of the virtual windtunnel.

References

[1] M.H. Smith, W.R. Van Dalsem, F.C. Dougherty, P.G. Buning, “Analysis and Visualization of Complex Unsteady Three-Dimensional Flows”, paper AIAA-89-0139, American Institute of Aeronautics 27th Annual Aerospace Sciences Meeting, Reno (1989).

[2] R. Haimes and M. Giles, “VISUAL3: Interactive Unsteady Unstructured 3D Visualization”, paper AIAA-91-0794, American Institute of Aeronautics 29th Annual Aerospace Sciences Meeting, Reno (1991).

[3] S. Shirayama, “Visualization of Vector Fields in Flow Analysis I”, paper AIAA-91-0801, American Institute of Aeronautics 29th Annual Aerospace Sciences Meeting, Reno (1991).

[4] W.J. Yang (editor), *Handbook of Flow Visualization*, Hemisphere Pub., New York (1989)

[5] J.P. Crowder, “Tufts”, *Handbook of Flow Visualization*,, pp. 125-175

[6] P.K Kundu, *Fluid Mechanics,* Academic Press, San Diego (1990), pp. 51-53.

[7] A. Globus, T. Lasinski, C. Levit. “A Tool for Visualizing the Topology of Vector Fields”. (submitted to *Visualization '91*).

[8] L.A. Yates and G.T. Chapman, “Streamlines, Vorticity Lines, and Vortices”, paper AIAA-91-0731, American Institute of Aeronautics 29th Annual Aerospace Sciences Meeting, Reno (1991).

[9] I.E. McDowall, M. Bolas, S. Pieper, S.S. Fisher and J. Humphries, “Implementation and Integration of a Couterbalanced CRT-Based Stereoscopic Display for Interactive Viewpoint Control in Virtual Environment Application”, in *Proc. SPIE Conf. on Stereoscopic Displays and Applications*, J. Merrit and Scott Fisher, eds. (1990).

[10] Fisher, S. et. al., Virtual Environment Interface Workstations, *Proceedings of the Human Factors Society 32nd Annual Meeting*, Anaheim, Ca. 1988

[11] D. Jespersen and C. Levit, “Numerical Simulation of Flow Past a Tapered Cylinder”, paper AIAA-91-0751, American Institute of Aeronautics 29th Annual Aerospace Sciences Meeting, Reno (1991).