Volume Rendering for Computational Fluid Dynamics: Initial Results

Samuel P. Uselton, Computer Sciences Corporation

1.0 Abstract

This work describes a direct volume rendering technique adapted for visualization of results from Computational Fluid Dynamics (CFD) flow solvers. The problem is difficult for at least three reasons: (1) the shapes of the grids, (2) vector as well as scalar data fields, and (3) the types of features of interest. In addition, the data set sizes are often between ten and one hundred megabytes, which is large even by volume rendering standards. The method, a ray casting implementation, is chosen for its robustness and adaptability. The software described here differs from other volume renderers in the handling of vector field data and in the techniques for reducing the time required for intersecting rays with grid cell faces. Vector fields are rendered using mappings from the vector field space to three dimensional color spaces plus opacity. Several mappings are considered and the relative merits analyzed. Two techniques for improving the speed of the ordinarily slow ray casting are described. Future developments and improvements are also discussed.

Introduction:

This work describes a direct volume rendering technique adapted for visualization of results from Computational Fluid Dynamics (CFD) flow solvers. Direct volume rendering is a term used to describe a collection of related techniques used to display densely sampled 3D volumes of data as clouds of varying color and opacity. The problem of direct volume rendering CFD data is quite difficult for several reasons. The grids used in computational fluid dynamics work can be severely warped, so algorithms for regular grids cannot be used. Velocity fields are of central interest, so vector as well as scalar data needs to be displayed. Features of interest include shock waves, flow separations and vortices. These structural features of the flow are difficult to isolate. In addition, the data set sizes are large, as much as one hundred megabytes for a single instant in time, which is large even by volume rendering standards.

An introduction to the problem of visualizing computational fluid dynamics (CFD) solutions is provided below. It is followed by a brief survey of visualization techniques, both those currently used for CFD and the previous uses of volume rendering. Then the current implementation, a ray casting method chosen for its robustness and adaptability, is described in detail. Several resulting images are presented along with discussions of the parameter values used to create the visualizations, and of features visible in the images. The results are followed by an evaluation of the current status of this project and a description of further work to be done.

Problem Description:

Fluid dynamics is an area of physics intensely studied due to the extreme complexity of the phenomena and the practical importance of the results. This field is concerned with the flow of fluids, generally within or around some solid object. Example applications include air flowing past an aircraft, water flowing past ship or submarine hulls, blood flowing through hearts and arteries, and fuel flowing through engines.

Traditional fluid dynamics experiments use wind tunnels, water tanks, or other complex apparatus or may be part of flight testing aircraft. Much time and effort goes into building models and assuring precise control of the apparatus. Mathematical models which describe flows have been used to compute results similar to the data collected from wind tunnels and water tanks. Over the last decade, computational fluid dynamics (CFD) has taken an increasing role in fluid dynamics research and engineering. While creating the digital description of a model is still a difficult undertaking, numerical simulation has several advantages. Once the geometric description is available, minor variations can be made much more easily than on a physical model. Parameter values can be varied through a greater range in a simulation than on real equipment, and there is no instrument apparatus to modify the flow being measured. Further, data values are available at a much denser sample of positions in the flow than is generally possible in physical experiments. The geometric nature of the problems and solutions as well as a visual tradition from the physical experiments has naturally led to an interest in computer graphics for displaying these results.

The results of these simulations are typically in the form of a three dimensional array of several values per location. For unsteady flows (flows that vary with time) each time step of the simulation creates another solution array. The geometric locations of the nodes corresponding to array values are stored explicitly in an additional three dimensional array. Each node, corresponding to an array element, is assumed connected to the adjacent nodes before and after it in each dimension of the array, for a total of six neighbors. This connectivity defines a grid of a volume of space. The volume is divided into cells with six faces each, so each cell can be thought of as a distorted cube. The collection of cells is known as a structured grid. Sometimes several of these arrays are created, each associated with a different region of the flow and a different part of the body in the flow. In this case, the separate grids may interpenetrate, and require special handling in the regions of duplication.

Sometimes the subdivision of the volume of interest is done in a less regular manner, resulting in a similar set of three dimensional points and associated values, but for which adjacency information must be kept explicitly. These “unstructured” grids contain mainly tetrahedral cells, but may have other shapes as well. An ideal visualization tool would have the capacity to display data of both the structured and unstructured grid formats.

Current methods of direct volume rendering are difficult to apply to CFD solution data for several reasons. The most obvious of these difficulties concern the grids on which the data samples are calculated and the type of features that researchers would like to observe. The size of the data sets further aggravates the problem of display, especially when interactivity is needed.

Direct volume rendering techniques were developed to operate on scalar data sampled or calculated on a grid of points arranged in a regular, rectangular pattern, which we will call a regular grid. CFD grids fail to meet this specification in several ways. The grids are designed to fit the volume of space adjacent to objects in the flow; these objects are often aerodynamic shapes, both grossly and subtly curved. Current computing resources limit the number of sample points that can be used, so care is exercised in placing the points. Samples are computed more densely in regions of greater expected complexity, for example the boundary layer, adjacent to surfaces of objects in the flow. As a result, the size of the cells and the spacing between points can vary by several orders of magnitude in the same grid. While the grids are indexed in a rectangular manner, the geometry is severely warped and compressed. Unstructured grids lack even the topological similarity to regular grids.

The most common use of direct volume rendering has been in the medical field. One of the reasons for this success has been the nature of the features of interest. Biological structures such as body organs, muscle tissue and bone, as sensed by Computed Tomography (CT) or Magnetic Resonance (MR) scanners, are homogeneous with fairly abrupt boundaries. Value ranges can be used to classify volumes and simple gradient operators are often effective for finding boundaries between structures of interest. In contrast fluid flow is basically continuous, and computational models which are built from

differential equations must approximate this continuity. So simply identifying the features of interest and extracting them is more difficult. In fact, it may not always be obvious what aspects of the data constitute “a feature”. This problem is compounded by data which represent vector fields or even tensor fields. Once the features are identified, the question of how best to render them remains.

Many scientific visualization applications indicate data set size as a difficulty. Even within this class of applications, CFD data sets are notable for the number of bits representing what one desires to see in a single image. The number of cells in a typical grid ranges from a minimum useful size of about 100,000 cells to grids of complete aircraft (a Harrier YAV-8B) with more than two million cells. These sizes are of at least the same size as those encountered in medical applications. However, since the grid is not geometrically regular, the spatial coordinates must be explicitly stored - three floating point values per cell. The typical “flow solver” calculates several values per cell. The standard solution file at NASA Ames has two scalars and one three dimensional vector per cell, all floating point values. Post processing frequently computes additional scalar and vector fields from these initial values, and while they may not require long term storage, they take workstation memory while being rendered.

For example, current work on modeling the F-18 has resulted in 4 grids averaging 400,000 cells each, for a total of 1.6 million cells. (This is just half the plane; the flow over the other side is assumed to be symmetric.) The F-18’s 1.6 million cells, with eight values per cell in double precision, requires roughly 100 megabytes. The bandwidth to simply copy this data, assuming rendering took no time, is huge. A small data set of 40,000 cells, (the blunt fin described below) frequently used for testing purposes, requires roughly 1.3 megabytes of memory. And these numbers are for a flow at a single instant in time. Unsteady flows may require hundreds or thousands of such solution files (although the geometry file usually remains constant).

Previous Work:

Previous and current (graphical) methods of exploring CFD data:

Physicists and engineers have, of course, long used traditional two dimensional graphs, function plots, contour diagrams and similar visual presentations of data. Aeronautical engineers and fluid flow physicists have an additional visual tradition in the use of dye in water tanks, smoke emitters, oil streaks and tufts of string in wind tunnels and during flight tests..

The initial applications of computer graphics technology were simply automating the same processes. Computers make creating graphs and plots of much larger data sets easier, while maintaining a high degree of accuracy. Tools to mimic wind tunnel visualization methods followed. Streamlines, streak lines and path lines are three variations on the smoke trail theme, which are available in current CFD visualization tools such as Plot3D [Wala90] and FAST [Banc90]. A streamline is a curve which is everywhere along its length tangent to the flow field at an instant in time. A streak line is the current location of all fluid particles that have passed through a fixed point for an interval of time. This set of

locations are the ones that would be colored over time by releasing dye from a single point. A path line is the trajectory of a massless particle released in a flow at a particular point. It can be thought of as a time exposure of a glowing particle in the flow. In a steady flow or a solution computed for a single instant in time these three items are identical, but in an unsteady flow they differ in interesting ways.[Kund90]

The use of the computer makes additional visualization techniques available. Psuedocolor display of a data values defined on a two dimensional surface has been done, much as in image processing. These surfaces may be the surface geometry of the airfoil, a layer of the grid defined by holding one index constant or an arbitrarily oriented cutting plane. Surfaces of a constant scalar value (a 3D generalization of contour lines) can be computed and displayed.

Recent work has shown the viability of additional techniques. For example neighboring streamlines can be connected to form stream surfaces which show relationships between the stream lines and direction changes in the flow much more vividly. [Hult?] Topological characterization of the flow is another recent development that shows much promise.{Helm90][Helm91][Glob91]

With the sole exception of topological characterization, all of these methods display only a small subset of the information contained in the flow solution in a single image. While the computation of a flow solution requires hours of CPU time on the fastest supercomputers, it is important to support the interactive exploration of the result. In this way the scientist can modify the selection of the subset displayed as well as the methods for displaying it. Interactively moving a cutting plane or changing the value of the desired isosurface allows exploration of the entire volume. These techniques, and others such as interactively moving the source of a stream surface, tremendously improve the speed with which a scientist or engineer can discover the information of interest in the data.

Direct volume rendering displays a dense set of 3D data in its entirety. It can be thought of as a complementary tool, showing the complete data set in a ways that highlight regions which are interesting targets for additional exploration with the other tools. Certain parameter settings may also reveal three dimensional structures that are not obvious when sliced by a two dimensional primitive.

Direct volume rendering:

Direct volume rendering was initially an alternative to the process of feature detection, feature extraction and surface reconstruction for data sets dense in three dimensions. These data sets came primarily from measurements of the real world. In medicine, the Computed Tomography scanner was followed by Positron Emission Tomography and Magnetic Resonance Imagery. The medical field also digitizes microscope images at various focal depths, and serial sections produced by slicing samples. Oil field explorers and other geophysicists have been making three dimensional seismic surveys for more than ten years.

Since the data was being collected by humans, it was collected in ways that made it convenient for further use. In particular, the samples were usually collected at regular

intervals and with a desirable alignment. Sometimes the step size in one dimension might be different than the step size in the other two, but even then it was generally fixed for that direction. The resulting data set is easily stored in a three dimensional array representing a regular grid of sample values. A starting position and an increment in each dimension make a complete description of the geometry.

Regular grids permit many kinds of optimizations in rendering. The fixed step size allows fast stepping from one cell to the next when tracing the path of a ray from the eye through the volume, accomplished by a three dimensional variation of the digital differential analyzer (DDA) algorithms [ ]long used for line drawing on raster displays [Fole90]. The same knowledge of unchanging distances permits easy weighting by the thickness traversed and precomputation of weights to use with neighboring samples in computing interpolated values [West89] [West90]. Assuming orthographic projection of the regular grid buys even more efficiency, since the projection of all the cells will be identical. [Wilh91] The data from any single sensed modality is often a single scalar value like CT density or amplitude of the seismic return wave. The resulting displays could be in greyscale like the X-ray images already familiar to radiologists, or psuedocolored as in the image processing of satellite remote sensing data.

The computer graphics research community put increased attention on direct volume rendering beginning about 1988. Three papers in that year's SIGGRAPH conference proceedings [Dreb88] [Sabe88] [Upso88] have been followed by special workshops on Volume Visualization in 1989 and 1990, as well as many other publications in journals of the field. [Levo89] Previous work on visualization of densely sampled volumes of data had concentrated mainly on finding geometric descriptions of surfaces within the volumes. [Lore87] [Wyvi86] Much was published explaining various shading and visibility attenuation methods and techniques for rapidly traversing the large data sets. Relatively little appeared, however, discussing similar techniques applied to densely sampled volumes of data for which no easy front-to-back (or back-to-front) ordering was available until the San Diego Workshop on Volume Visualization in December 1990. Papers presented there described methods for displaying unstructured grids and badly shaped structured grids such as those arising in CFD and finite element analysis. [Gari90] [Hanr90] [Shir90] The work described below is designed for these deformed grids and in addition attacks the problem of displaying vector fields over these densely sampled volumes.

Description of method:

Direct volume rendering techniques attempt to compensate for large data sets by exploiting the coherence present in grid structures. Since CFD grids are scaled and twisted severely, there is no simple indexing of the grid that can assure back-to-front or front-to-back traversal of the volume. Further, the irregularity of the individual cells means that DDA extensions for rapid stepping through the grid also fail. The obvious technique robust enough to function in these circumstances is ray casting (ray tracing of only the initial rays, originating at the eye), accumulating shading contributions as the ray traverses

the grid until the ray leaves the grid vicinity or until the shading can not be further influenced.

Ray tracing is a technique widely used in computer graphics for creating realistic appearing images.[Fole90] Basically, ray paths from the eye through sample screen points are followed into the scene until an object is encountered. Rays from the point of intersection to the light sources and in the directions of principle reflection and refraction are followed to compute shading contributions for the screen sample point. Reflected and refracted rays can be pursued until the contribution of each individual ray is below the color representation threshold of the display system. While reflected and refracted rays used in realistic image synthesis are one of the hallmarks of ray traced images, the reflections and refractions generated can be distracting and confusing when attempting to analyze scientific data. Eliminating these rays also reduces the work to generate each image. Attenuation of light through a translucent medium can also remove the need to follow a ray beyond a certain point, since no light penetrates so far.

The present implementation makes use of two strategies for accelerating the normally slow ray casting process. One idea exploits the facts that cells are always closed and that they share faces. The other idea allows rapid finding of the first cell hit.

A naive (volume rendering) ray tracing implementation considers each cell or each face of each cell as a separate object. Each ray is tested for intersection with each object. The nearest object intersected is employed in a shading calculation, and then the search is performed to find the next intersection. The coherence of this grid may be exploited by observing that the cells are always closed and that cells share walls except at grid boundaries, which are easily identified by the index values (or by explicit adjacency information in unstructured grids). Therefore, once the cell nearest the eye hit by a particular ray has been identified, the subsequent cells penetrated by the ray can be found, in the order traversed, by finding the nearest intersection among the five remaining walls of the cell already entered. The next cell entered is simply the one which shares that wall. If the newly hit wall of the cell is also a wall of the grid, then the ray has exited the grid. It is possible for the ray to re-enter the grid after exiting so an explicit check for this possibility must be made.

To find the first cell hit by each ray, we adapted the item buffer technique described in [Wegh84]. The item buffer technique uses fast, hardware supported Z-buffer rendering, but stores an object identification number, rather than a color, in the frame buffer. In this way, the first object hit by a ray is identified without expensive ray-object intersection testing. In our case, we encode the (i, j, k) index of each cell into an RGB value. As long as each dimension of the grid is less than 255 elements, the mapping is unique, and trivially invertible. The grid cells are rendered by a Z-buffer using the (i, j, k) to RGB encoding. For each ray we get the RGB value at the corresponding pixel. The inverse mapping (from RGB to (i, j, k)) is performed, which yields the starting grid cell. Note that if the viewpoint is sufficiently close, some of the grid cell walls are clipped by the hither plane. Then interior cell walls may be the first cell wall visible, so all cell walls are drawn, rather than just the outer shell of the grid. The window used for the Z-buffer drawing is also used for

selecting the view of the data set to be volume rendered. By drawing only the outer grid walls until the view selection is satisfactory, the view selection can be done interactively.

These techniques are not sufficient to allow interactive ray casting, but there is hope that with a more powerful workstation and further optimization techniques, this goal may be achieved. Some strategies with good potential for reducing the time required are discussed in the Further Work section, below.

Vector Field Rendering:

Standard techniques can be used for mapping scalar field values to color ranges for use in shading calculations. [Sabe88], [Dreb88], [Upso88] Much of what CFD researchers wish to see is found in the vector fields. The standard output at NASA Ames includes a momentum vector, which, when divided by density at each cell, yields the local velocity of the fluid. The Plot3D package currently used for visualization by many CFD researchers includes seven vector functions (in addition to more than fifty scalar functions) any of which might be candidates for display by volume rendering. Each field is useful in its own way and which ones are appropriate depends on characteristics of the flow such as compressibility, turbulence and viscosity and on the particular features of interest in the flow field.

Vector fields are traditionally rendered as collections of arrows which encode magnitude as length, direction as orientation of the major line segment, and location as the point at the tail of the arrow. This representation is adequate for a two dimensional field whose representation elements are sufficiently far apart to permit drawing of parallel arrows that do not touch. A more densely represented field can be rendered by using a subset of the actual field. Integral curves, streamlines for example, are an example of this strategy. A surface in a three dimensional field can also be treated this way, at the risk of ambiguity in the directions. Vector fields on a surface can also be rendered by mapping direction to hue and magnitude to saturation, lightness or opacity. This technique is more appropriate for very densely sampled fields and fields in which one wishes to show interpolated values between samples. Both these techniques fail if the vector field is densely sampled in three dimensions.

Volume rendering has the potential to display dense three dimensional vector fields by using the multidimensional nature of color spaces. Most displays using higher dimensional color spaces are difficult to interpret. The challenge lies in finding “good” mappings from the desired vector field(s) to a color space. A “good” mapping is one which permits a scientist to gain insight into the “interesting” aspects of the vector field. “Interesting” depends on the scientist and the research area.

Vortices in the velocity field are of particular interest to aircraft designers. Rapid local variation of the pressure field is indicative of a possible vortex, but is not conclusive evidence. A high magnitude of vorticity is necessary but not sufficient evidence. The vortical structure is more apparent in the vector fields of velocity and vorticity. We decided to attempt to construct a good mapping for displaying vortical structures.

Our initial attempt uses the dimension of opacity as well as the dimensions of the Hue-Lightness-Saturation (HLS) color space to present vector fields. HLS color space was chosen over the display system's natural Red-Green-Blue (RGB) space because intermediate values in the HLS system are more intuitively understood by most people.

Opacity is a very powerful cue so we want it to be used to represent the most important dimension of the data. Interesting regions should be more opaque; less interesting regions should be more transparent. The magnitude of the vorticity vector serves this function well, but magnitude of the velocity vector does not. The uninteresting parts of the velocity field are the areas far from the objects, where the velocity is equal to the free stream velocity (the velocity that would be present if there were no obstructions), and on the surfaces of bodies, in the flow. A fluid with non-zero viscosity sticks to any surface, implying zero velocity (relative to the surface) everywhere on the surface. This condition is known as a no-slip boundary condition.

Subtracting the "free stream" velocity from the velocity field at every cell yields the perturbation velocity. If the magnitude of the perturbation velocity vector is mapped to opacity, the bulk of irrelevant free stream flow is invisible. The no-slip boundaries require separate treatment.

Given the use of vector magnitude as one of the dimensions, the most appropriate representation of the vector fields is in polar coordinates, as a magnitude and two angles. This representation requires designation of an axis, to be used as the zero direction vector. We selected the free stream velocity direction, because it represents the direction of undisturbed flow. The two angles necessary are then an azimuth, or rotational, angle around the free stream vector and an elevation, or upstream to downstream, angle. The rotational angle around the free stream direction is mapped to the obvious circular dimension Hue. The alignment is arbitrary, and we have used the y axis direction as zero. The elevation angle the field vector makes with the free stream vector is mapped to saturation, so that vectors almost parallel with free stream are nearly gray (unsaturated) and perturbation velocities of 180 degrees to free stream are maximally saturated. Additional variations of this mapping are the subject of future experiments.

Additional Implementation Details

The background for the volume rendered images is a gridded surface perpendicular to the viewing direction. The visibility of the grid lines gives useful visual feedback about the degree of opacity at each pixel. This technique was suggested by Peter Shirley, in his presentation of [Shir90] at the San Diego Volume Visualization Workshop.

Several different attenuation formulas have been used on an experimental basis. The most realistic formula for the effects of partial opacity of a medium involves exponential attenuation of the light as a function of distance traveled through the medium. This formulation has proved quite useful in volume rendering regularly gridded volumes. Realism, however, is not the ultimate goal in scientific visualization. Realism is a subgoal, valued in scientific visualization mainly because it usually improves the ease of accurately interpreting the resulting images. In CFD grids, the small cells are small specifically

because the scientists believe that something interesting is likely to be in the vicinity, and they want to see it in greater detail. Not only is the increased sample density needed for display of finer detail, but more fundamentally,smaller cells are required to correctly calculate the flow solution. The small cells in a CFD grid are often the most important ones. Reducing their impact on the image due to their size is counterproductive. The attenuation formula I have found most useful for revealing details in CFD flows is one that values all cells equally, regardless of size. This evaluation is actually simpler, since it can be done separately for each cell wall, without knowledge of the distance between cell wall intersections. All of the images in the following section use this model.

Results:

The data set used for most of the testing of this prototype software implementation is the blunt fin data set described by Hung and Buning [Hung84]. This data set is used because it is small enough to be easily held in workstation memory and yet complex enough to contain interesting features. In the original work, several psuedocolored grid planes are shown for two scalar functions, pressure and local Mach number. The images contained in the original paper have served as a useful guide in selecting scalar fields to volume render, in designing color mappings for them, and as an important verification tool.

The volume of interest is a region of flow past a blunt fin which rises out of a flat plate. (See Figure 1.) Since the flow is assumed to be symmetric about the center line of the fin, the flow solution is computed for only half of the volume surrounding the fin. Two faces of the structured grid coincide with the surfaces of the plate, shown in red, and the fin the straighter yellow grid. A second yellow grid plane is also shown as an example of a typical grid plane of that orientation. A third face of the grid is defined by the plane of symmetry, shown as the closest blue grid. The last grid plane of this orientation and a typical one are also shown in blue. The other three faces of the grid are simply “far enough” out from the fin and plate and “far enough” downstream from the leading edge of the fin to permit an accurate solution by the flow solver. Note the increasing density of the grid lines as they approach the two surfaces.

The flow is parallel to both the plate and the length of the fin, at slightly less than three (2.95) times the speed of sound. Both the plate and the fin are specified as no-slip surfaces, so the velocity at the surfaces is zero. The grid is 32 nodes by 32 nodes by 40 nodes for a total of 40,960 nodes. At each of these locations, the density, momentum and energy of the flow have been computed. From these values many others can be calculated. For example, the velocity at each node is the momentum vector divided by the density, and pressure is easily calculated from the velocity and density. The calculator module of FAST [Banc90] was used to calculate several fields of interest for this and other data sets used in testing.

Figures 2 through 6 are presentations of various scalar fields of this flow, using a variety of color transfer functions. In general, the same scalar value has been used to control both the hue and the opacity of the volume in its neighborhood. The values used at each intersection of a ray with a cell face are calculated by bilinear interpolation between the four cell face vertices. All transfer functions are piecewise linear interpolations from the data range to the visual parameter range. The colors used in these images have been

changed from the ones used on the CRT screen because colors are perceived differently in the two media. On the screen of a light emitting device like a CRT, high intensity produces bright colors which stand out, and they stand out particularly well against a dark or black background. Printed copies of these images use large amounts of ink to produce the dark regions and little ink for the light colored regions. The dark regions attract more attention, and the paper surrounding the image is almost always white. For this reason, the colors have been inverted before being printed. The white backgrounds with black grid lines are, on screen, black backgrounds with white grid lines. Bright red, when inverted, becomes cyan (an intense turquoise); green becomes magenta; and blue becomes yellow. Figure two is presented in both forms to emphasize this point. A brief discussion of each image highlights some visible features and describes the parameter values used.

In figure 2, the density values are mapped to both opacity and color range. The color range designed for the CRT has red as the maximum and blue as the minimum. Green, the other monitor primary is halfway between these extremes. Yellow, a mixture of red and green, falls between red and green in value; cyan, lying between green and blue, results from values in the corresponding data range. The high density just in front of the leading edge of the fin is visible in red, fading through yellow and decreasing in opacity as distance from the fin increases. The inverted colors used for paper output have cyan as the maximum, yellow as the minimum, and magenta as the halfway value. The same high density features can be seen, but detail is more easily picked out in the inverted color image. For example, the major trailing shock wave can be seen as a slightly more opaque region further from the fin as one looks downstream.

In figure 3, the local Mach number is mapped to a similar color range and opacity function. The distinct, double branched “lambda” shock is easily visible in cyan near the base of the leading edge of the fin. This is one of the features described in [Hung84].

Helicity is calculated as the dot product of velocity and vorticity. It is a scalar value which has a large magnitude near vortices. Its sign may be either positive or negative and indicates the direction of rotation of the vortex. For figure 4, the image of helicity, large magnitude is opaque regardless of sign, and small magnitude is nearly transparent. Extreme negative values are mapped to cyan, extreme positive to magenta, and values near zero are mapped to yellow. Two vortices starting near the base of the leading edge of the fin can be seen. The tight cyan vortex stays close to the fin and rises slowly. The magenta vortex moves both out from the fin and up from the base plate more rapidly.

The velocity field is the object of central interest in the study of fluid flow. In this case, the x component is parallel to the unobstructed flow. The presence of the fin in the flow generally reduces this component of the velocity. In fact there are some negative x component values in the data set. These areas are of particular interest because reversal of the flow direction is often indicative of a separation of the flow from the surface of the body. Flow separations are also features of interest to aircraft designers. In figure 5 the smallest x values (including the most negative) are mapped to cyan and high opacity, while the largest x velocity components are mapped to yellow and near transparent.

The perturbation velocity field is defined to be the result of subtracting the free stream velocity for the data set from the velocity field at each point. The x component of this data set is parallel to the length of the fin. Negative values of the perturbation velocity x component represent locations in the field where the flow is slower than the free stream velocity. Since low values are most interesting, they have been mapped to cyan and opaque, as in Figure 5.

The same transfer function is used to create the images of figures 5 and 6. This transfer function uses the extreme values of the data field being presented to set the scale and offset into the linear color space. The results are very similar even though the numerical values are quite different. In both images the cyan flow reversal region is visible.

The next three figures are the results of initial attempts to volume render vector fields. In each case the free stream velocity vector is used as an axis for determining the mapping from the parameter of interest into visual parameters. This mapping uses a spherical coordinate system rather than a Cartesian system in order to create a more natural mapping into a multidimensional color space. The coordinates in this spherical system are magnitude, azimuth and elevation (or magnitude, longitude and latitude). The magnitude of the vector is mapped to opacity. The radial angle around the free stream vector (azimuth) is mapped to hue (color) because hue has a natural range of 360 degrees. The angle the vector makes with the free stream vector (elevation) is mapped to saturation, the dimension that ranges from gray (at zero saturation) to full brightness.

The first attempt at vector field volume rendering is a direct attempt to display the velocity field. The blunt fin geometry is clearly visible in dark reddish brown. The main shock wave in the flow is visible as the division between the forward area which seems to oscillate between green and magenta hues, and the rear area which is mainly green. However, the oscillations in the forward region are very distracting and reflect more on the grid and the flow solver than on the flow solution of interest. It has been suggested that the oscillation may be an artifact of a slight instability in the numerical methods of the solver, exaggerated by the particular color mapping. For small magnitudes, the direction may be almost random, resulting in a wide variation in hues even though almost transparent.

We selected the perturbation velocity field as a good candidate for volume rendering before discovering that aerodynamicists had a particular name for it. Basically, the reasoning is to make the uninteresting part of the field near zero and the interesting part far from zero. Then mapping the magnitude of the vector to the opacity becomes a natural way to select the interesting parts of the field. Choosing spherical coordinates and the mapping from these coordinates to visual parameters was more difficult. In figure 8 CRT colors are used because they are clearer in hardcopy than the inverse colors ordinarily used. The blunt fin geometry is again visible. Since the perturbation velocity everywhere on the no slip boundary is exactly the negative of the free stream velocity, the vector has a relatively large magnitude and zero azimuth angle. The result is nearly white walls for both the fin and the base plate, with opacity relatively high due to the (relatively) large magnitude. The shock wave and vortex structures are visible although the saturation is low enough that the images are not very compelling. The vortex that hugs the fin is purple. The main shock is yellow-green against the reddish main flow. The red direction is “up” from

the plate (zero hue which matches the y axis) and yellow-green is away from the fin. A legend for clarifying which hue is associated with which direction would enhance the value substantially.

Another vector field of interest is the vorticity field. Vorticity is the cross product of the gradient of the velocity field with the velocity field. Vorticity is always high in the boundary layer, due to the large change in velocity over a small distance. It is also high in the neighborhood of vortices, where the direction of flow changes rapidly. Because we wish to see the vortices but not the high vorticity boundary layer, a different transfer function must be used. Figure 9 is an attempt to display the vorticity field. More work is needed on this transfer function. In addition to modifying thresholds in each dimension of the mapping, it would probably also be useful to determine a different axis to use for the spherical coordinate system. Since vorticity is always at right angles to the local velocity vector, it is often nearly at right angles to the free stream. Therefore saturation is almost always near 50% and provides little information.

Evaluation:

Direct volume rendering is one of the few methods of displaying large, densely sampled sets of data that provides a view of the entire data set in a single image . For this reason, I believe it has good potential to earn a place in the suite of tools used for developing an understanding of complex flows. Most other tools require a subset of the flow to be selected for inspection at any one time. This selection must be made before seeing the data, although it can usually be interactively modified after seeing a previously selected subset. Having a general view which highlights areas of complexity can provide important information about what subset would be interesting to observe in more detail and what kinds of tools might be most appropriate for the further analysis. As indicated by the images in the previous section, direct volume rendering can also be useful in displaying large scale structures of a flow.

The usefulness of the images is quite sensitive to several decisions best left in the hands of the researcher, who knows the most about the data and the features of interest. First, the particular field to be displayed, and the ranges of values in that field which are of particular interest, should be carefully selected. Once these decisions are made, a set of transfer functions can be designed which do highlight the desired ranges. This task is not too difficult for scalar data values and can be made much easier by the use of appropriate auxiliary tools, such as histograms. Even so, care must be exercised to avoid confusions between regions of intermediate data values and visually superimposed regions of two distinct colors.

Opacity is a particularly valuable parameter for highlighting subranges and indicating presence while deemphasizing the importance of other ranges. Hue is also a strong cue, except for color-blind viewers. It should be used for an important data range, but a natural mapping can be difficult to find.

People are not, in general, accustomed to thinking of three dimensional color spaces so interpolations between points in a three dimensional color space produce unexpected

results. The non-intuitive nature of these results make the difficult problem of interpreting the data even worse. For these reasons, constructing appropriate transfer functions is much more difficult for vector valued fields. Using a color space that is more easily interpreted by people rather than the one that is natural for computer graphics display systems is important. Changing the coordinate system of the vector field to improve the geometric match with the color space is also helpful. Even so, while it is clear that “something is happening” in particular regions in the volume, analyzing what that “something” might be is still quite difficult.

Direct volume rendering is unlikely to displace other methods of graphically exploring CFD data sets. Other techniques, such as particle traces and stream surfaces are likely to be superior in displaying fine detail in the structure of complex flows. There are trade-offs in the size of the ranges of interest and the visual complexity which also limit its utility. If the range of interest is limited to a single value, and made opaque, while all other data values are completely transparent, the result is an iso-surface. Increasing the width of the range while decreasing the opacity results in a thickening of the surface to a volume and making its boundary appear less solid. Increasing the opacity of the “other” regions slightly further reduces the solidity, making the regions appear fuzzy.

Further Work:

The primary objection to use of the current software is the time required to generate a final image. The window used for the volume rendered image is set at 512 x 512 pixels. Times to generate the volume rendered images with one ray per pixel vary from 10 to 30 minutes. The resulting image must be worth considerable study if the user is to wait that long for it to be generated. There are several places where the software can be made more efficient, which should result in reducing the time requires by at least half. Five to 15 minutes is still too long to wait for changes in viewpoint or changes in the transfer functions.

A second window provides a view of the grid rendered using the Silicon Graphics Z buffer hardware, which allows interactive selection of position and orientation of the data set. In this way, at least the images generated are likely to be desirable ones. Alternative strategies for radical reductions in time to produce a volume rendered image are being pursued.

The usefulness of the images produced varies substantially and one of the most important factors is the mapping from data values to visual parameters. Improving the speed would help by making comparisons of images computed from slightly different mappings easy to accomplish. Integrated tools for manipulating the mappings are an important feature to be added soon. Two different aspects need to be supported. The first is interactive shaping and editing of the transfer function which maps the input data value range to the output visual parameter range, like the stand alone module provided by Shankar Ramamoorthy of University of California at Santa Cruz. Having some information about the distribution of the input data is critical to designing a good transfer function. This observation was the motivation for my writing the histogram tool kit. [Usel91] Kristina Miceli has begun a small project to combine the functionality of these two tools. Eventually something similar should be accessible from within a volume renderer. Once such tools are available,

more exhaustive experiments to evaluate various transfer function designs and attenuation formulas are in order.

Another important aspect of the colorization of the data is the ability to interpret the result. Providing a legend or key to the display would improve the interpretability substantially. For scalar fields, a bar showing the sequence of colors and labeled with the extreme data values would be a major improvement. The form of the key, however, is far from obvious, for the multidimensional mappings used for displaying vector fields. One possibility is to show the free stream direction arrow, (or whatever axis is used for the spherical coordinate system) a plane containing the full range of hues at elevation of ninety degrees, and a small number of semicircles of equal longitude, with the appropriate colors mapped.

Most of the figures show rendering artifacts which are the result of numerical inaccuracy in the ray following, bad interpolation values or other similar errors. Improving the speed of the code will permit more test runs and better debugging.

Adding the ability to perform the same sort of rendering for data defined over unstructured grids is easy in principle. The changes required are mainly a matter of adding an alternative data structure for the storage of the data values and geometry. The ray intersection and shading routines are fundamentally the same, except that the number of faces exiting a cell is in general smaller.

Another useful feature would be an option to allow the rendering of two dimensional geometry in the same universe as the volume. The obvious first application would be to display the geometry of the solid that is deforming the flow. Cutting planes, clipping planes and similar probes would also fit into this modification.

Bibliography:

[Banc90] Bancroft, G.V., F.J. Merritt, T.C. Plessel, P.G. Kelaita, R.K. McCabe, and A. Globus, “FAST: A Multi-Processed Environment for Visualization of Computational Fluid Dynamics,” Proceedings, Visualization ‘90, IEEE Computer Society Press, pp 14-27.

[Blin82] Blinn, James F., “Light Reflection Functions for Simulation of Clouds and Dusty Surfaces,” Computer Graphics, vol 16, no 3, (July 1982) pp 21-29.

[Dreb88] Drebin, Robert A., Loren Carpenter and Pat Hanrahan, “Volume Rendering,” Computer Graphics, vol 22, no 4, (August 1988) pp 65-74.

[Fole90] Foley, J. D., A. van Dam, S. K. Feiner and J. F. Hughes, Computer Graphics: Principles and Practice, 2nd ed. Addisson-Wesley, Reading, Mass., 1990.

[Garr90] Garrity, Michael P. “Raytracing Irregular Volume Data,” Computer Graphics, vol 24, no 5, (November 1990) pp 35-40.

[Glob91] Globus, A., C. Levit and T. Lasinski, “A Tool for Visualizing the Topology of Three-Dimensional Vector Fields,” Proceedings of Visualization ‘91 Conference, Oct 1991.

[Good89] Goodsell, David S. and Arthur S. Olson, “Molecular Applications of Volume Rendering and 3-D Texture Maps,” Proc, Chapel Hill Workshop on Volume Visualization, May 18-19, 1989, pp 27-32.

[Hanr90] Hanrahan, Pat, Nelson Max and Roger Crawfis, “Area and Volume Coherence for Efficient Visualization of 3D Scalar Functions,” Computer Graphics, vol 24, no 5, (November 1990) pp 27-34.

[Helm90] Helman, J.L. and L. Hesselink, “Surface Representations of Two- and Three-Dimensional Fluid Flow Topology,” Proceedings, Visualization ‘90, IEEE Computer Society Press, pp 6-13.

[Helm91] Helman, James L. and Lambertus Hesselink, “Visualizing Vector Field Topology in Fluid Flows,” IEEE CG&A vol 11, no 3, (May 1991) pp 36-46.

[Hu 89] Hu, X., K.K Tan, D.N. Levin, S.G. Galhotra, C.A. Pelizzari, G.T.Y. Chen, R.N. Beck, C-T. Chen and M.D. Cooper, “Volumetric Rendering of Multimodality, Multivariable Medical Imaging Data,” Proc, Chapel Hill Workshop on Volume Visualization, May 18-19, 1989, pp 45-49.

[Hult90] Hultquist, J. P. M.,“Interactive Numerical Flow Visualization Using Stream Surfaces,” Computing Systems in Engineering, v 1, no 2-4, pp 349-353, 1990.

[Hung84] Hung, C.-M. and P.G.Buning, “Simulation of Blunt-Fin Induced Shock Wave and Turbulent Boundary Layer Separation,” AIAA Paper 84-0457, AIAA Aerospace Sciences Conference, Reno NV, January 1984.

[Kaji84] Kajiya, James T. and Brian P. Von Herzen, “Ray Tracing Volume Densities,” Computer Graphics, vol 18, no 3, (July 1984) pp 165-174.

[Kund90] Kundu, Pijush K., Fluid Mechanics, Academic Press Inc., San Diego, CA, 1990.

[Levo88] Levoy, Marc, “Display of Surfaces from Volume Data,” IEEE CG&A, Vol 8, no 3 (May 1988), pp 29-37.

[Lore87] Lorensen, William E. and Harvey E. Cline, “Marching Cubes: A High Resolution 3D Surface Construction Algorithm,” Computer Graphics, vol 21, no 4, (July 1987) pp. 163-169.

[Sabe88] Sabella, Paolo, “A Rendering Algorithm for Visualizing 3D Scalar Fields,” Computer Graphics, vol 22, no 4, (August 1988) pp 51-58.

[Shir89] Shirley, Peter and Henry Neeman, “Volume Visualization at the Center for Supercomputing Research and Development,” Proc, Chapel Hill Workshop on Volume Visualization, May 18-19, 1989, pp 17-20.

[Shir90] Shirley, Peter and Allan Tuchman, “A Polygonal Approximation to Direct Scalar Volume Rendering,” Computer Graphics, vol 24, no 5, (November 1990) pp 27-34.

[Upso88] Upson, Craig and Michael Keeler, “VBUFFER: Visible Volume Rendering”, Computer Graphics, vol 22, no 4, (August 1988) pp 59-64.

[Usel91] Uselton, Samuel P., “Histogramming Tool Set with Application to Visualization of CFD Data Sets,” RNR Technical Report RNR-91-006, NASA Ames Research Center, January 1991.

[Wala90] Walatka, Pamela P., Pieter G. Buning, Larry Pierce and Patricia A. Elson, “PLOT3D User’s Manual,” NASA Technical Memorandum 101067, March 1990.

[Wegh84] Weghorst, Hank, Gary Hooper and Donald P. Greenberg, “Improved Computational Models for Ray Tracing,” Trans. on Graphics, vol 3, no 1,(Jan 1984) pp 52-69.

[West89] Westover, Lee, “Interactive Volume Rendering,” Proc, Chapel Hill Workshop on Volume Visualization, May 18-19, 1989, pp 9-16.

[West90] Westover, Lee, “Footprint Evaluation for Volume Rendering,” Computer Graphics, vol 24, no 4, (August 1990) pp 367-376.

[Wilh90] Wilhelms, Jane, Judy Challinger, Naim Alper, Shankar Ramamoorthy, Arsi Vaziri, “Direct Volume Rendering of Curvilinear Volumes,” Computer Graphics, vol 24, no 5, (November 1990) pp 41-48.

[Wilh91] Wilhelms, Jane and Allen Van Gelder, “A Coherent Projection Approach for Direct Volume Rendering,” Computer Graphics, vol 25, no 4, (July 1991) pp 275-284.

[Wyvi86] Wyvill, Brian, Craig McPheeters and Geoff Wyvill, “Data Structure for Soft Objects,” The Visual Computer, vol 2, no 4, (1986) pp 227-234.