

Math4205
Project Report:

By: Ashley Dive,

33549860

Andrew Hill,

40279361

Due Date: 29th of
October, 2004

Topic:

Comparison of different visualisation techniques applied to fluid dynamics

Introduction:

Visualisation Programs used:

Fluent - particle advection, streamline
OpenDX - isosurface rendering
Matlab - volumisation rendering

The problem was given to us by Mathew Brennan from JKMRC (Jullius Kruttschnitt Mineral Research Centre) and is a cyclone simulation. Mathew provided a fluent case and data file which had reached satisfactory convergence. Cyclones are used to separate high density material from low density materials. A slurry flow which is a combination of water and large particles was injected into the cyclone at approximately 4m/s. Most of the high density material flows around the wall at a very low velocity and is eventually ejected out of the bottom of the machine at approximately 2m/s. Very low density material, i.e. air, is concentrated in the centre, forming an air column and is ejected from the top at a velocity of approximately 10m/s.

Methods used:

Fluent: Native flow visualisation

OpenDX: Isosurface, tried to use the particle trajectory (data exported from fluent to native DX file)

Matlab: Imports from any DX native file into a vector list which is then rendered by matlab

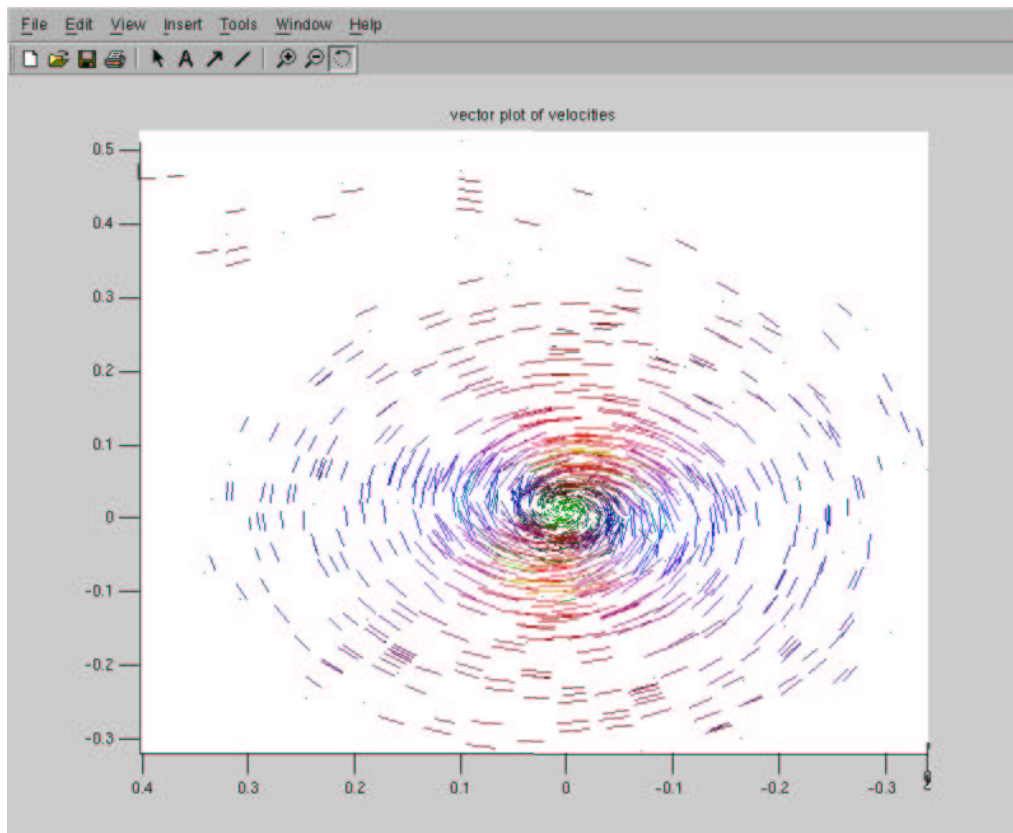
Matlab:

Reads a generic format OpenDX file, imports a list of positions then connections. Then the three following data sets are read, combined vector velocity (x,y,z, vector data) and 2 partial fractions (scalar data, floating point). The m-file was written in such a way that all variables are stored as floating point numbers which allows for all different types of data to be read and stored accurately.

A plot of vectors is produced, displaying every 100th point. The vectors are represented by a line with an arrow head pointing in the direction of fluid flow, with the length proportional to the velocity at that point. The image produced is similar to a streamline representation which is often used in CFD visualisations.

Unfortunately, Matlab is very inefficient at rendering single lines and hence only every 100th point is used. The range of values for the line length is [-1, 1] in the x, y and z direction, which gives a maximum magnitude of $\sqrt{3}$. Three colours are used, R, G, B. Each represents x, y and z velocities respectively.

Despite plotting only every 100th point, the visualisations produced were still high quality with a good trade off between line density and rendering speed. As this data set is irregular, choosing to render only a portion of data gives a good random distribution of visualised points. If the dataset was regular, the rendering scheme would have produced strong aliasing artefacts. The following image is representative of any vector visualisation produced by the m-file:



This shows a top down view of the entire dataset. The concentration of vectors is at the center as expected, as most material is drawn down toward this point.

The m-file can also convert the data along with position into volumetric data. Each volume element or voxel are tetrahedrons, with each of the 4 points defining a position in the volume. The points within each voxel were sorted by velocity and then the set of voxels were sorted by velocity. This resulted in all the voxels being sorted by the minimum velocity in each voxel.

The colour defined for the voxels is the same for the vector plot described above. The colour scheme is not the best model to use as the outside of the dataset has a velocity close to 0 (this represents the fluid flow against the walls of the cyclone). This gives a black outer skin on a complete volume render of all the voxels. We used Matlab's patch command to render the tetrahedrons as sets of triangles. The subset that was chosen to be rendered was the set of all voxels with a threshold greater than a specified value. This resulted in an isosurface at the specified velocity. An animated sequence which shows the rotation of the volume was produced. Due to the inability of Matlab in Linux to save compressed avi files, generation of any movies was impossible in the linux lab.

Rendering of these sequences on the windows machines presented technical issues due to the lack of swap space on the machines and the resulting large files. The avi file was rendered in five separate files and then stitched together. Further visualisations within Matlab were impossible due to the lack of native support for non-regular datasets. In particular we were unable to get Matlab's native stream3 function (which displays a standard 3D streamline) to work correctly as the function requires regular data.

OpenDX:

Reads a generic format OpenDX file which was exported from fluent.

A network was created which:

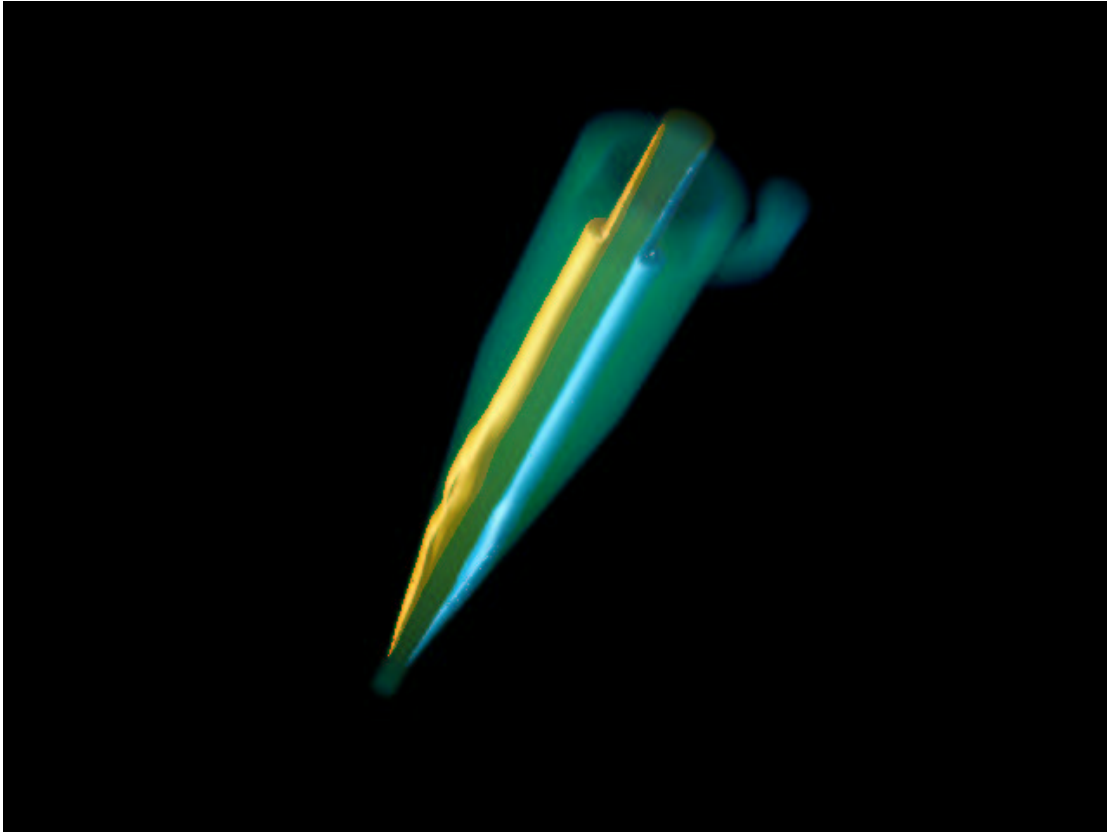
- 1) Displays an isosurface sequence beginning at the 0 isosurface and executing 60 times through to the value of 10.
- 2) Displays a combined isosurface and volume render to allow the viewer to compare the isosurfaces against the full volume.

Using a streamline would have been an excellent way to view the data, unfortunately OpenDX's streamline module cannot render the data effectively as the data is non-regular. Re-gridding the data takes over 10 minutes on a quad processor 2.4ghz xeon, which is very restrictive if the user only wants to investigate large scale flow geometries. The following image displays an isosurface at a particular value within the animated sequence:



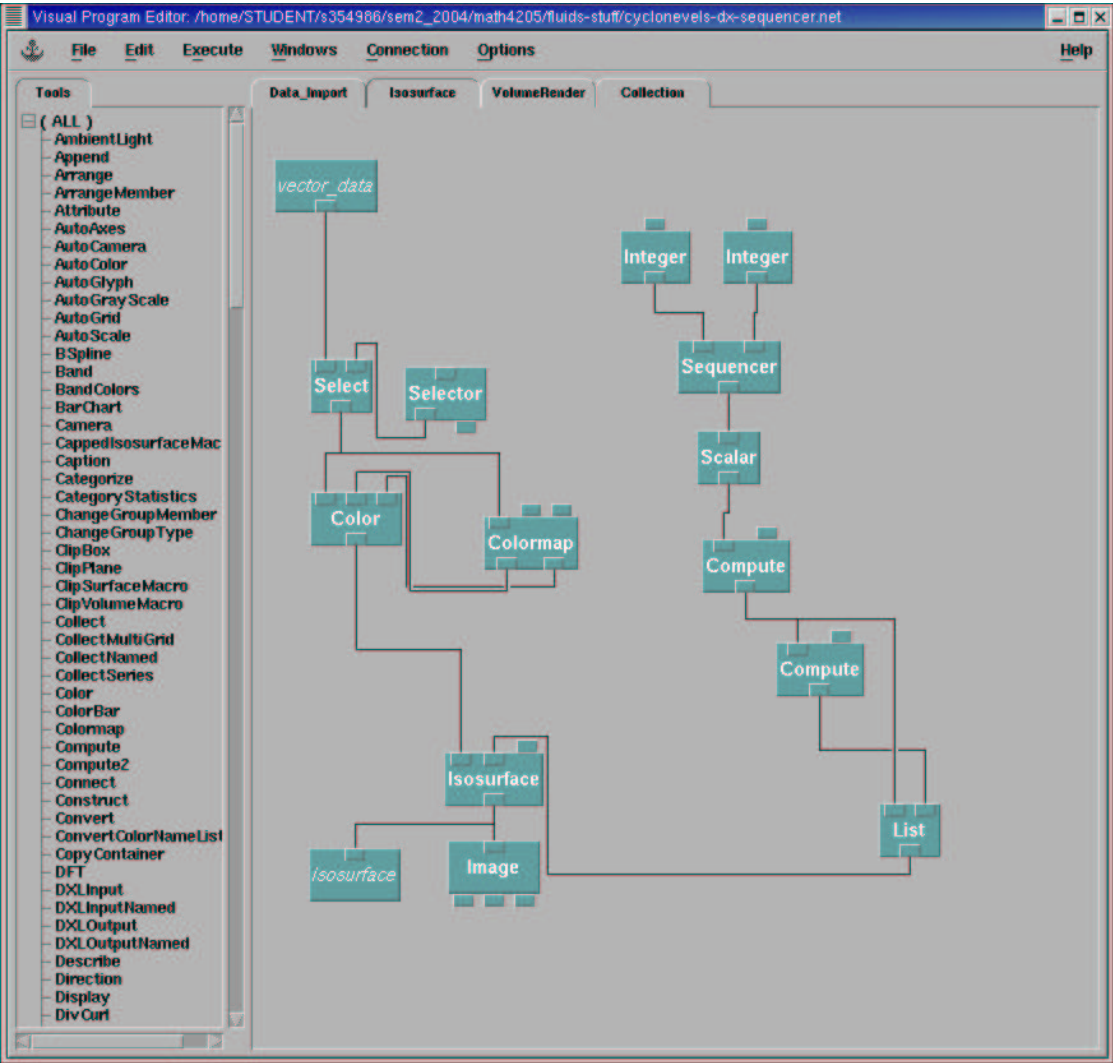
In this image, a non-symmetric twisting motion can be seen on the lower left of the isosurface. As the cyclone itself is symmetric, this shows how non-linear CFD work is.

The next image displays a combined isosurface and volume render of the same data and time step:



This image shows how useful combining volume rendering with isosurfaces can be. Here the viewer can easily see at what position within the volume the x-velocity isosurfaces are.

The animation was produced by the following network file:

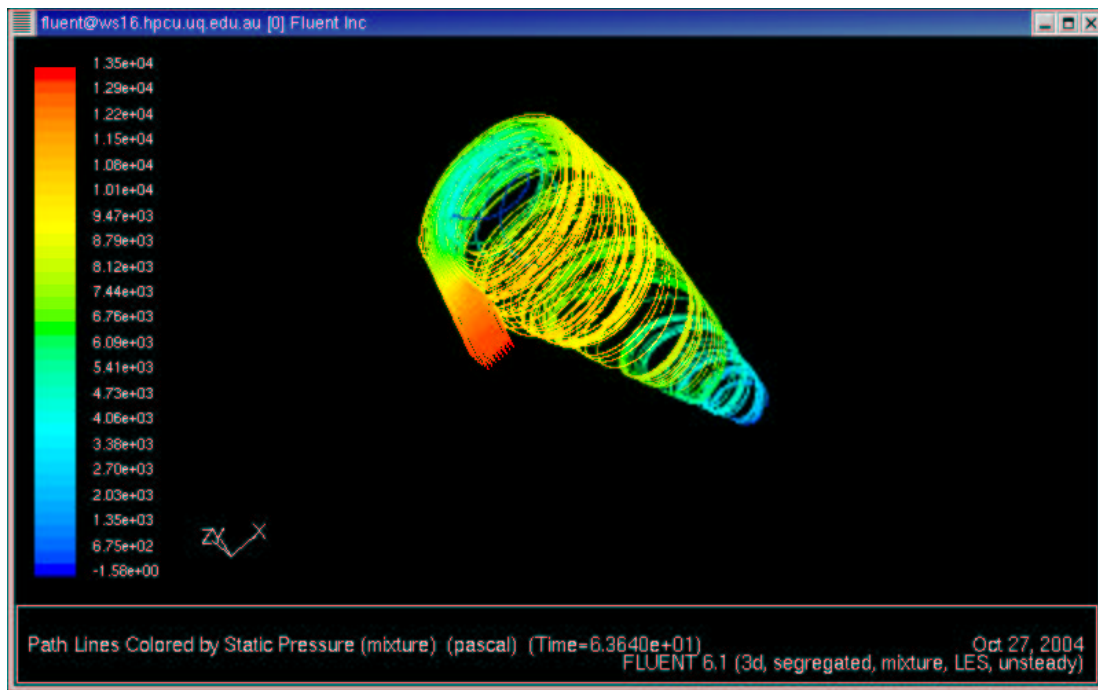


As any single direction in velocity has both a positive and negative value, the use of two computes was necessary to have both values plotted at the same time.

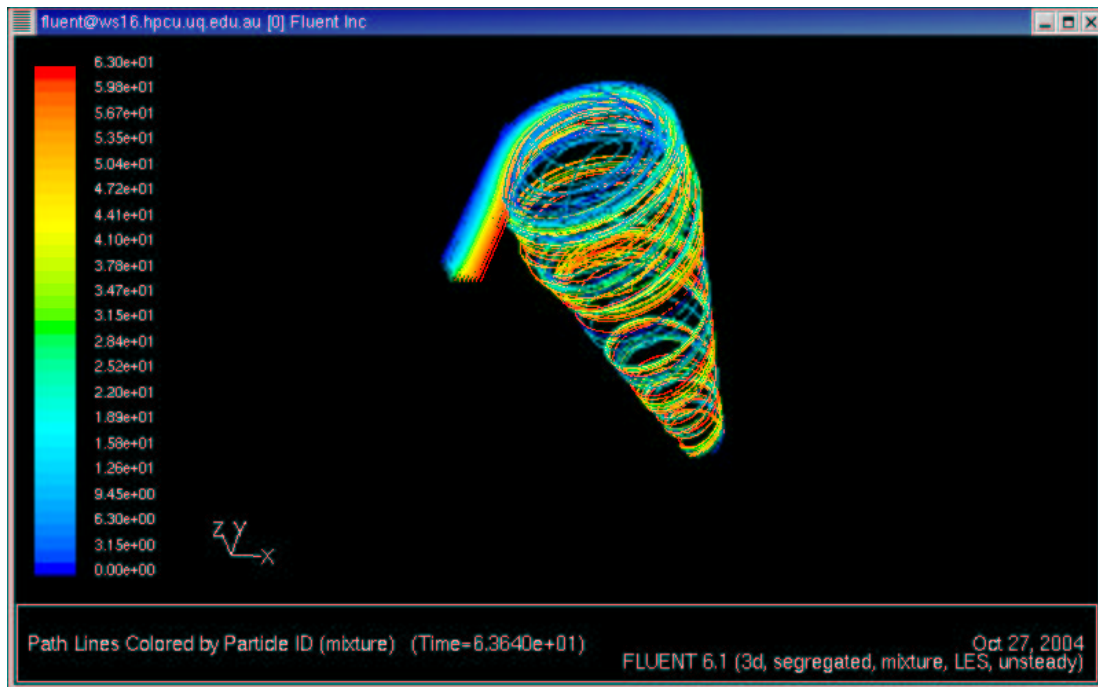
Fluent:

Fluent is a generic CFD (computational fluid dynamic) solver, it also includes a visualisation package which allows users to view the flow through systems they have produced. It solves 2d/3d, steady/unsteady states, in a discretised volumetric space. A mesh is loaded into the program, fluid properties defined (i.e. viscosity, or boltzmann simulation for non-cohesive particles), solver selected (i.e. 2d/3d, steady/unsteady), simulation run, data and cases saved then finally the flow is visualised.

Flow visualisation is preformed by selecting which variable/s are to be displayed and the shape/type of the particles. The visualisation component of fluent also allows a user to select the number of time steps to evolve the flow over and the size of the steps taken on each time step. This gives sufficient control to the experienced user to allow for the investigation of specific conditions. The following image displays the particle trace by static pressure:



The following image displays each trace colour as a particle ID:



Fluent does require some technical savvy to use and also a rudimentary understanding of the components and constants in CFD simulation. The number of variables which can be displayed is quite large, ranging from Velocity to Wall Fluxes to Turbulence. Fluent does provide some basic visualisation components for isosurfaces and static vectors.

Comparison between all 3:

Fluent has an excellent flow visualisation component. You can insert particles into the flow and watch the evolution of system as the particles move through it. Fluent does provide an interface to export data into many formats which are useful for flow visualisation, i.e. OpenDX, netcdf etc. Fluent is the best software available for doing any sort of fluids dynamics problems. There are multitudes of add-ons for it, i.e. acoustic simulation software. Support for multiple processors is available, though the costs are prohibitive. Using fluent does take time to master. Having multitudes of options and variables to change does pose a problem to the casual user, but the fluent documentation and tutorials is very comprehensive.

Matlab is not useful for volumetric or vector visualisation. Doing any sort of transparencies is a very difficult task due to the high level of difficulty in setting the opengl state engine into the right mode. Matlab should only be used for manipulating data and for solving small to medium sized mathematical problems. Any sort of visualisation should be done outside of Matlab by exporting the data into a useful format, ie. DX or binary data files. Matlab has many of add-ons available for it and it provides a java-machine parsing engine to interpret m-files. Any type of add-on can be added to the existing repertoire by learning Matlab's syntax which is similar to C, Fortran and Java. Matlab's license is not very expensive compared to other products i.e. Mathematica or Fluent, but it does not allow for multiprocessor computing.

OpenDX is the free software of choice for linux users and can be used via cygwin in windows. It provides a module interface so a user can program their own modules to do specific computations or manipulations. It has multiprocessor support as standard though it is not available in the VISAC labs. The time taken to render large volumes of data is enormous and hence is a non-realtime visualisation system. By rendering both isosurfaces and volumes together into the same image allows a user to reference where an isosurface is located. DX's volume rendering produces a very dull image due to the lack of data density in most volumes.

Conclusion:

The combination of using Fluent to view flow trajectories and OpenDX's isosurface and volume rendering methods would allow a CFD user to visualise their system and investigate any inconsistencies or anomalies. Matlab's inbuilt graphics subsystem is not flexible enough or fast enough to be considered as a candidate for performing visualisations.

An automated method for exporting Fluent data to an OpenDX format, reading and rendering a sequence of evolving isosurfaces would be an excellent tool for any CFD user.

We would recommend a combination of using both Fluent and OpenDX to visualise and investigate flow properties of any fluids topic.