CS 530 Final Project Report

Luke Jiang (0028440468)
05/02/2020

Dataset

I'm using the train dataset provided by the professor. I accessed the dataset through this link: https://www.cs.purdue.edu/homes/cs530/projects/data/final/cfd/train/.
The dataset describes the velocity, pressure and jacobian of wind travelling through a train.

Objective

The objective of this project is to explore the relationship between the pressure and velocity fields of the dataset. Specifically, the project implements a mechanism to draw a 2-D plot of velocity and pressure along a line specified by the user.

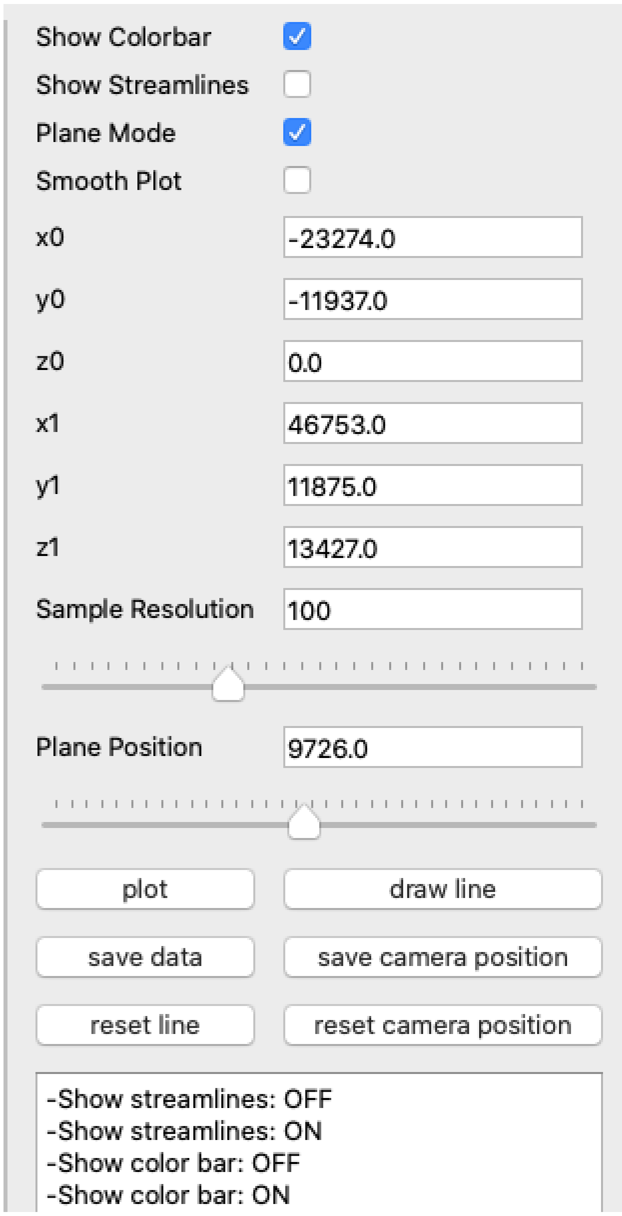
Approach

I first used paraview to explore the dataset. I found out that the most interesting region of the dataset is relatively small compared to the size of the whole dataset, so I preprocessed the dataset by clipping away nonessential parts using clip filters in Paraview.

After I have clipped the dataset to a reasonable size, I used vtkXMLUnstructuredGridReader to read the dataset in vtk. After the dataset is read, I used two pipelines for the main rendering. The first pipeline colored the data set with its pressure value and used a vtkCutter to implement a cutting plane, which gives the user a clearer impression on how the pressure value is distributed on that plane. The second pipeline used the same streamline techniques as in Project 4 to show some streamlines around the head of the train, where the pressure and velocity change the most. I colored the streamlines based on their velocity magnitude values.

I then proceeded to implement the plotting functionality. In order to display a line on the rendered result, I used vtkLineSource, which takes two points specified by the user. I then used vtkPointLocator to find out the pressure and velocity values at each point along the line. After that, I used vtkTable and vtkContextView to draw the 2-D plot and display the plot in a new window. The plotted curve may not be smooth, so I also implemented a 1-D low pass averaging filter to smooth out the curve, if the user would like to. For each point i in the sampling line, the filter uses (d[i] + d[i-1] + d[i+1]) / 3 as the averaged value of i.

Finally, I implemented a user interface to make it easier for the user to manipulate the rendering result as well as printing out other useful information including camera position and average value along the sampling line. The user interface looks like the following:



(Figure 1: user interface)

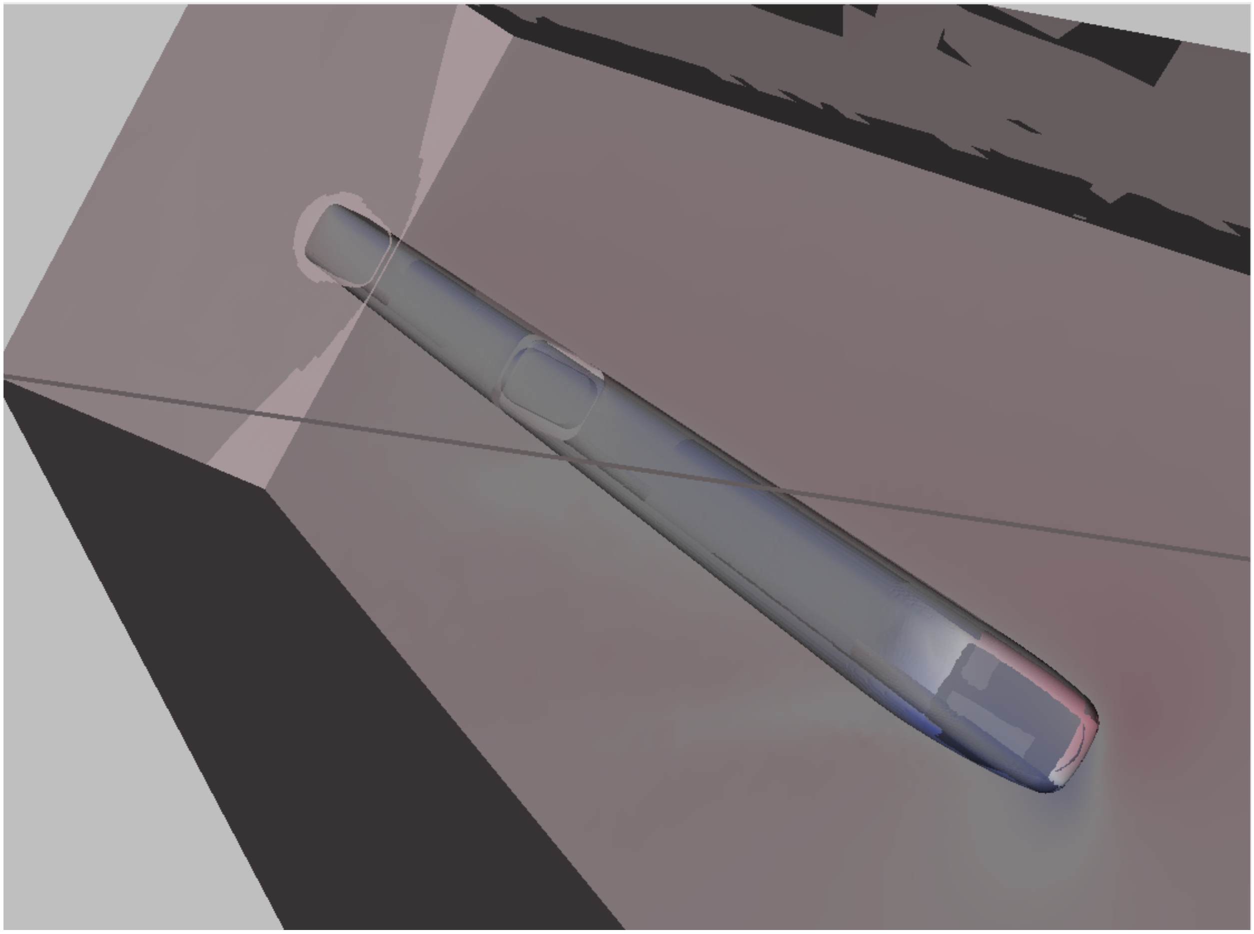
The top of the user interface contains two checkboxes that give the user the option to hide the streamlines and the colorbar. The “Plane Mode” checkbox provides a cutting plane that cuts across the train, and the user can see the pressure value on that plane more clearly. The “Smooth Plot” checkbox specifies if a smoothing effect is enabled. Below the checkboxes are six coordinates to specify the sampling line. The user can change the position of the line by typing new coordinates using the QLineEdit boxes. The user can also change the sample resolution using the slide bar. The “plot” button plots the 2-D graph. The “draw line” button is used to update changes of the coordinates and reflect the change on the rendering result. The “save data” button saves the velocity and pressure data into a file. The “save camera position” button saves the current position of the camera, which can be reset by clicking the “reset camera position” button. The “reset line” button resets the sampling line to its default position. A log section is also included to help the user keep track of the history of operation.

Problems

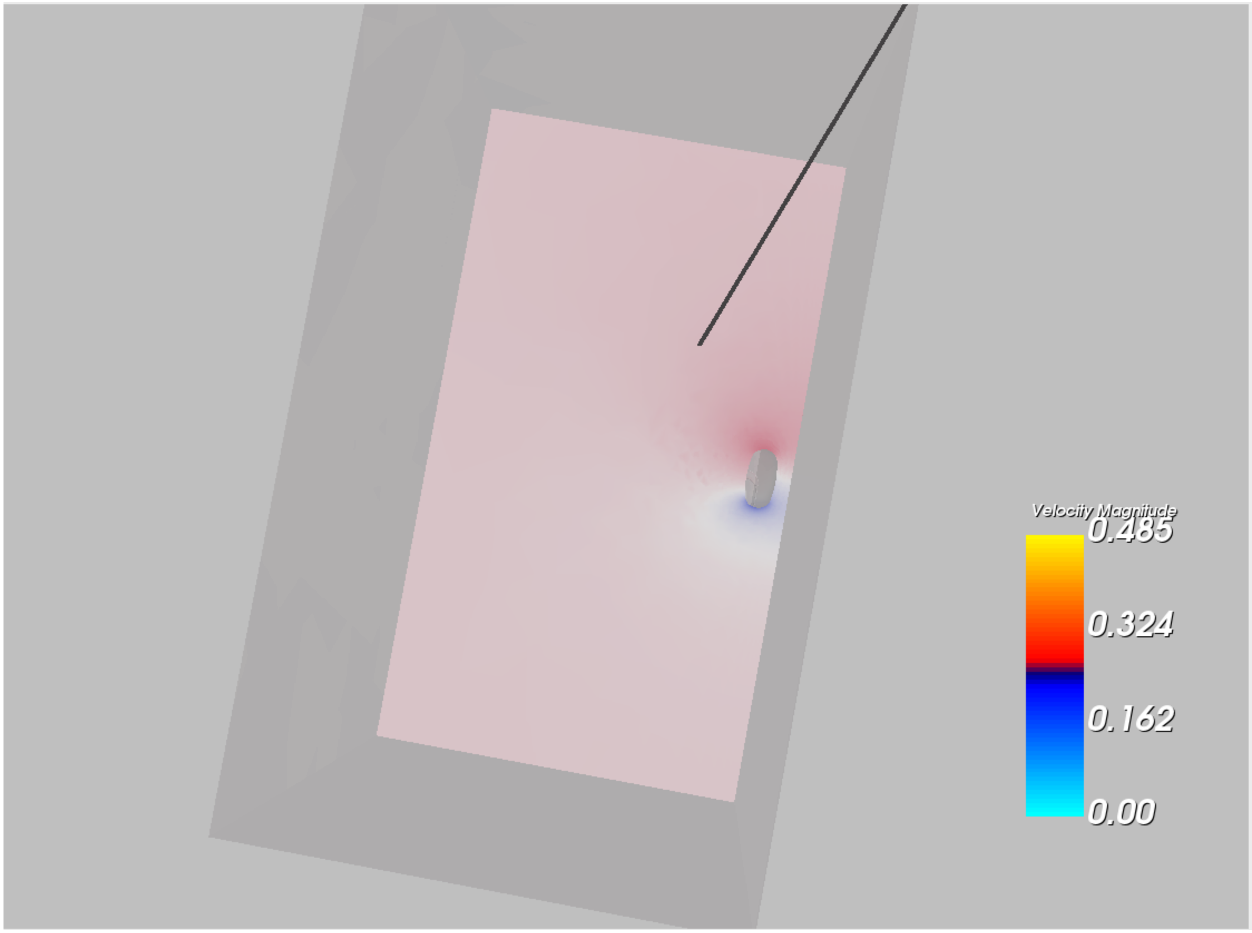
The biggest problem that I ran into was choosing the appropriate dataset to visualize. My initial proposal uses a different dataset, which cannot actually work with my old proposal. Therefore, I was behind my schedule when I was trying to find another dataset to work with. I also had a hard time making sense of the information contained in the dataset and what I can do with that information until I was advised to use Paraview to explore the dataset. Since I was way behind the schedule, I decided to do something easier to make sure that my project is at least complete.

The major difficulties I faced while implementing the project is how to implement the sampling function. I searched online for multiple ways to find the value at a specific point. I also had some minor difficulties with using the vtkContextView and vtkTable for plotting, and got help on Piazza. I also spent a lot of time redesigning the user interface as I was adding more and more features into the project.

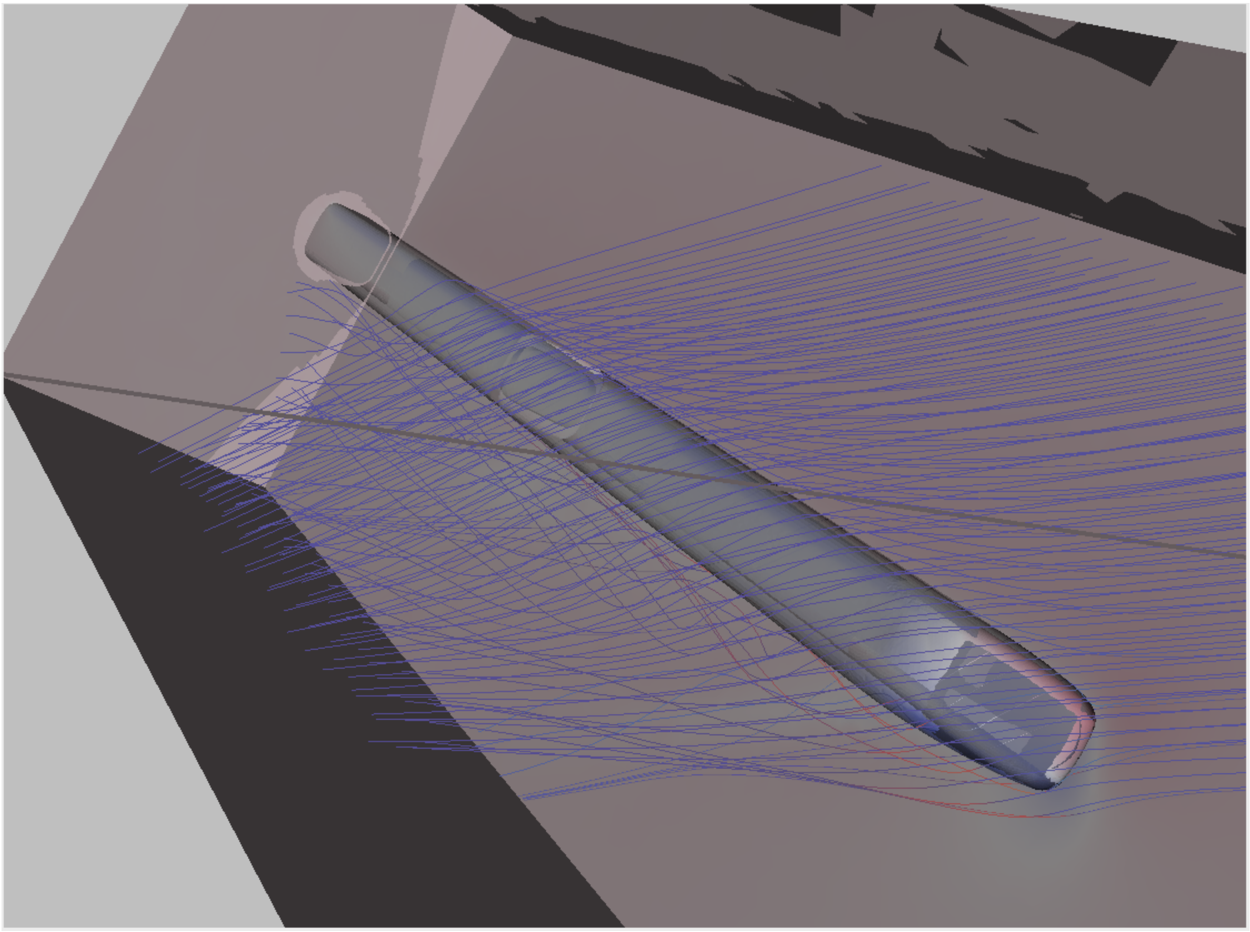
Results

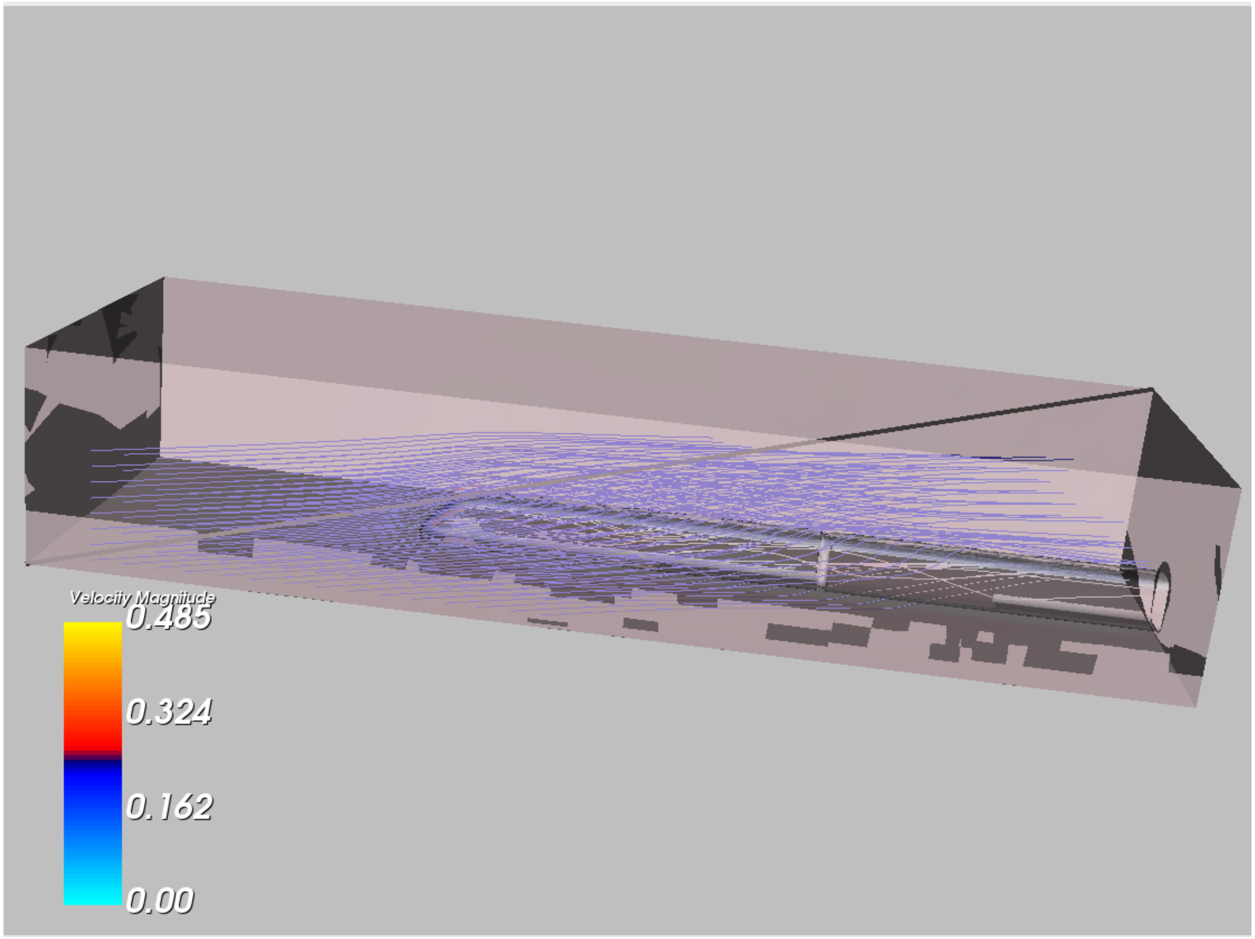


(Figure 2: pressure around the head of the train)

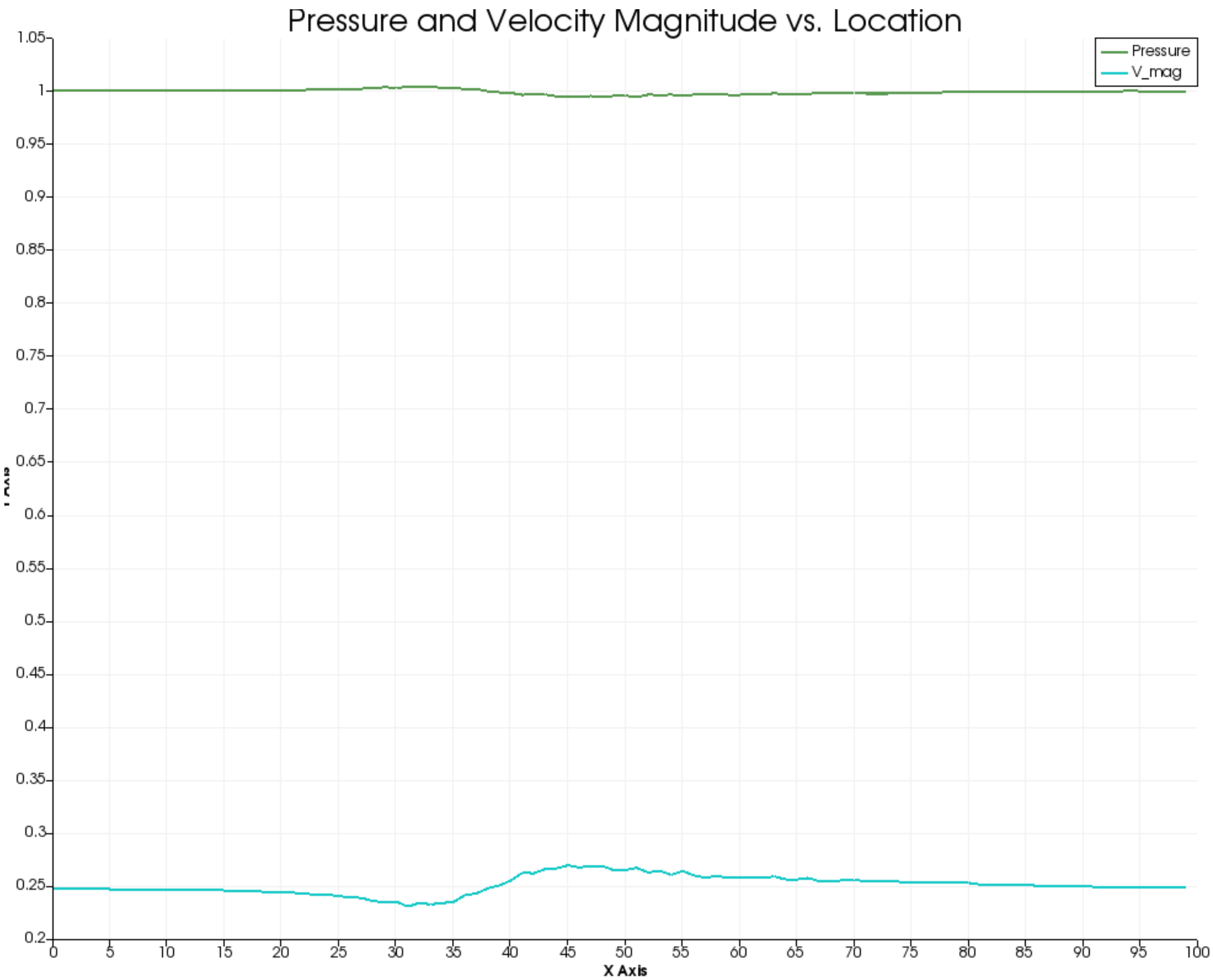


(Figure 3: pressure value near the head of the train using plane mode)

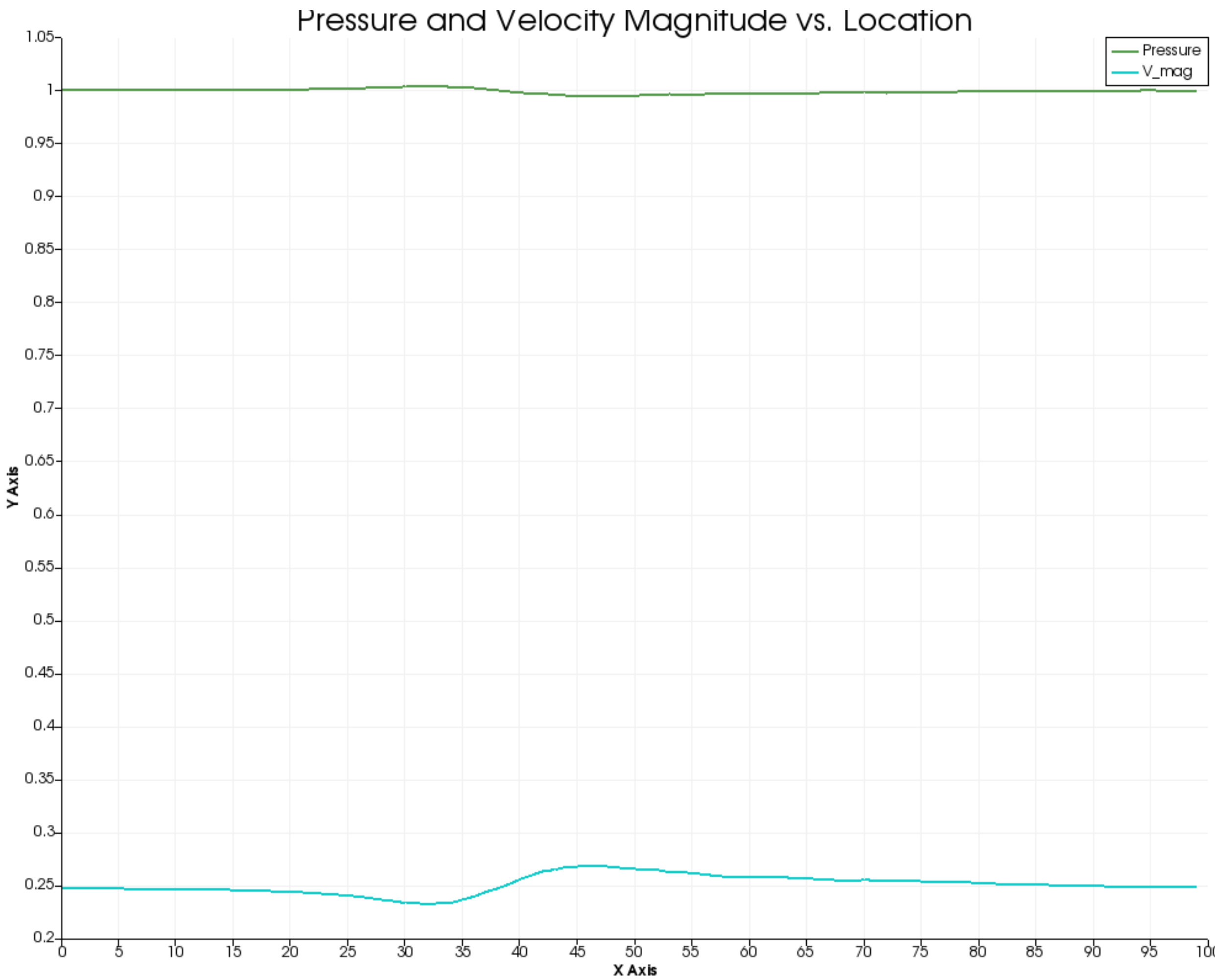

(Figure 4: streamlines)



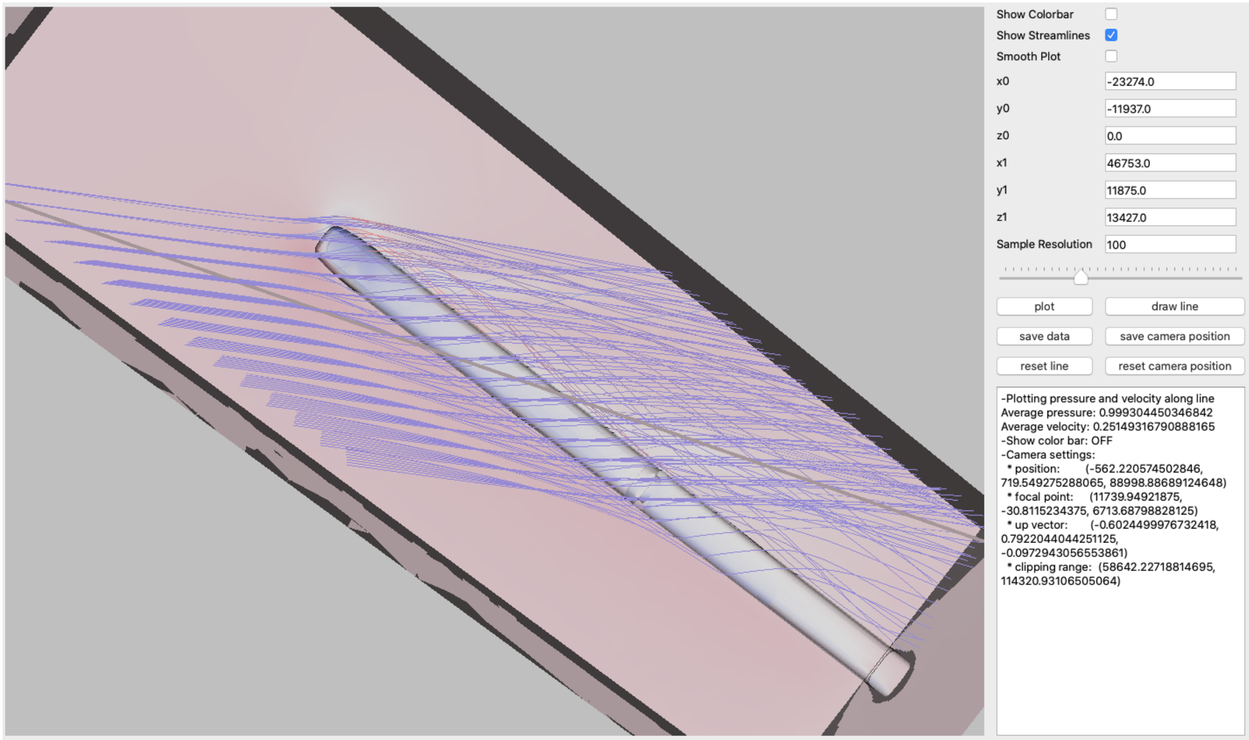
(Figure 5: initial sampling line and color map)



(Figure 6: pressure and velocity magnitude plot, unsmoothed)



(Figure 7: pressure and velocity magnitude plot, smoothed)



(Figure 8: overall program)

From figure 2 and 3, I discovered that the pressure of the left side of the train, which is colored in red, is higher than the pressure of the right side of the train, which is colored in blue. I think this is because the wind is blowing towards the left side of the train. From figure 2, 3 and 4, I observed that the values of pressure and velocity are related, because the colors of both fields change around the head of the train. Figure 5 shows the color map and the initial sampling line, which lines across the dataset. Figure 6 and 7 show the exact value of pressure and velocity magnitude along the sampling line. From these two figures, we can see that the pressure and velocity magnitude are indeed inversely related: The higher the velocity is, the lower the pressure would be. Figure 8 shows the overall layout of the user interface.

Conclusion

There are still a few drawbacks of my project. For instance, the color map around the head of the train is not obvious enough. The font sizes of the title and labels of the color map are disproportionate. There are also some extensions that I was planning to incorporate into this project but didn't have a chance to. I thought it would be beneficial to have two clipping planes that clips the dataset into only the part with the head of the train. In this way, the user can focus on the details of the interesting part of the dataset. However, the program took too long to respond after I implemented the clipping planes, so I decided to use a plane Cutter instead. I was also planning to have more plotting options, including plotting the velocity in the x direction only. However, I didn't have the time to do that.