Homework 4

Milena Belianovich
Wednesday, March 19

1 Part 1: Streamline Visualization of Air Flow above Heated Disk using Glyphs

- Adding the “Glyph” filter, setting the ”Scale Mode” to the vector magnitude, turning down the scale factor to reduce clutter, changing the color to ”Vectors.” Then changing the ”Glyph Mode” to ”All Points.” Figure 1 shows the underlying vector field without any resampling.

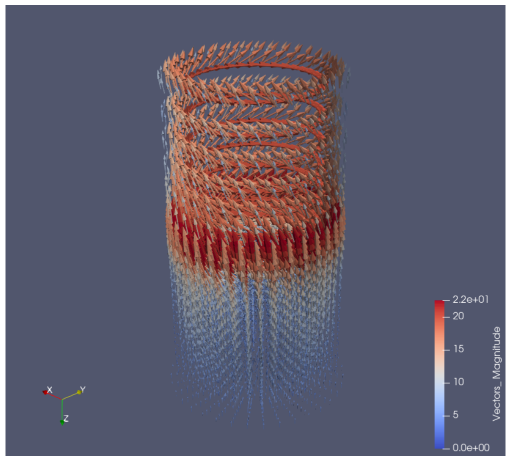

Figure 1: Part 1: glyph representing the underlying vector field

- Find and apply the appropriate filter to:
  - Extract streamlines using ‘Point Source” seed type.
    Along with extracting streamlines, we also add a proper color scheme as seen in Figure 2.

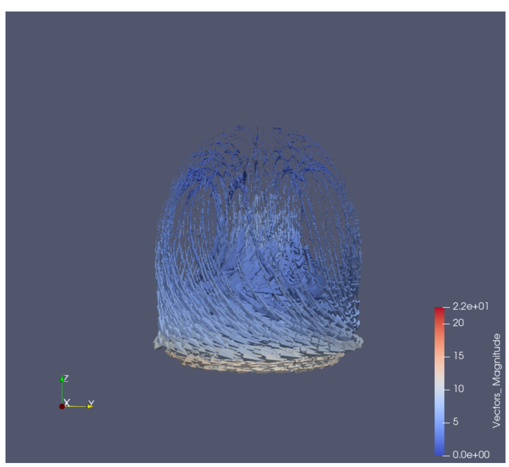

Figure 2: Part 1: streamlines extraction

code: https://github.com/milenabel/CS6635-VisforScD

– Enhance the rendering by using tubular surfaces.

Now, we turn down the opacity of the initial file as seen in Figure 3, and then enhance the rendering through tubular surfaces as seen in Figure 4.

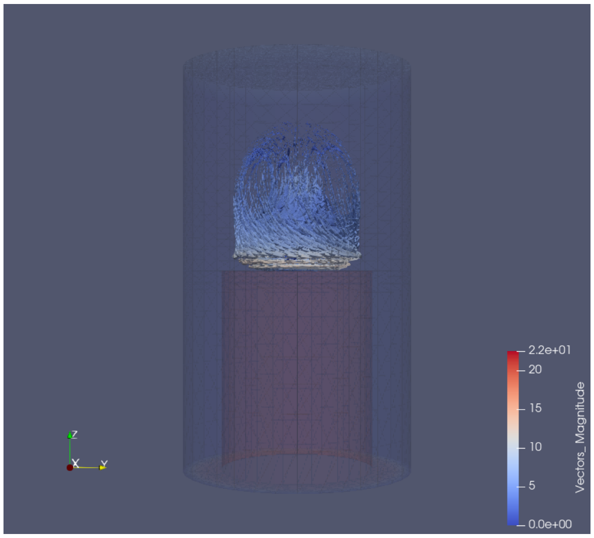

Figure 3: Part 1: streamlines with the initial plot (changed opacity)

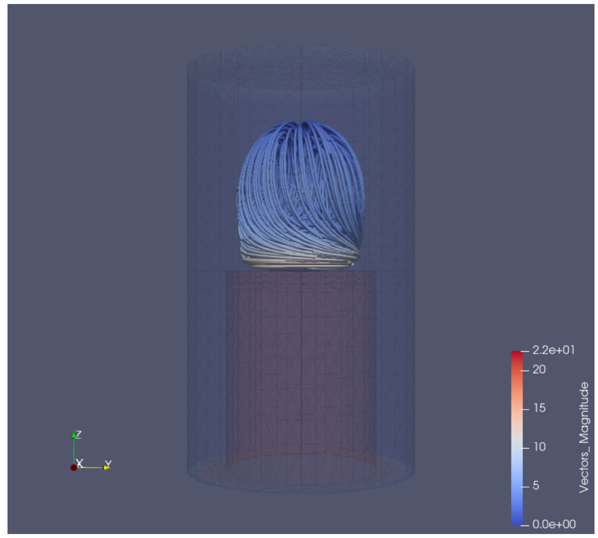

Figure 4: Part 1: streamlines enhanced by tubular surfaces

- Find and apply the appropriate filters to provide visual insights about the orientation and magnitude of the vector field in this region of space.

  After tweaking some "Stream Tracer" filter parameters, I have made it clearer to see areas of flow acceleration or deceleration, observe how the orientation correlates with the direction of the flow, as seen in Figures 5 and 6.

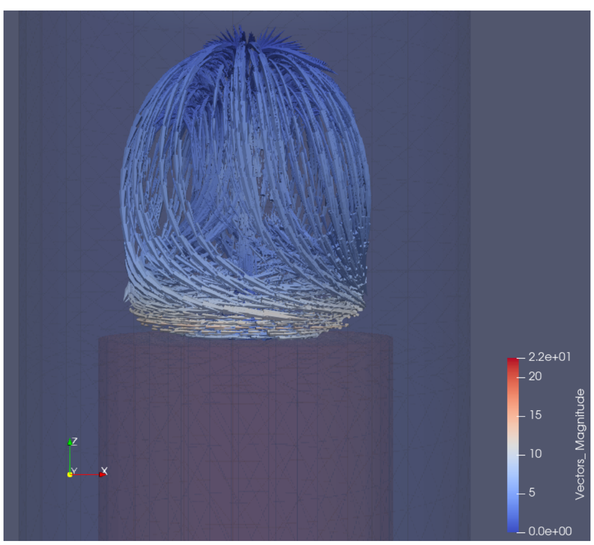

Figure 5: Part 1: insights 1.

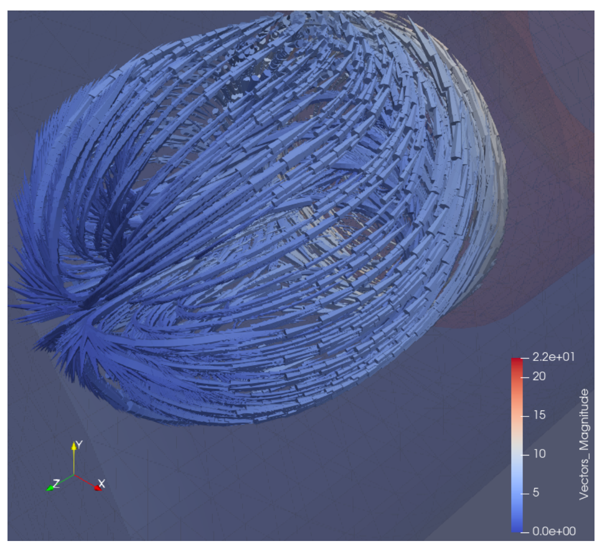

Figure 6: Part 1: insights 2.

code: https://github.com/milenabel/CS6635-VisforScD

2 Part 2: Hurricane Visualization

- Step 1:
  - Create streamlines of the wind flow within the hurricane. Seed the streamlines so as to get a good overview of the flow.
  - Create stream tubes of the wind flow within the hurricane.
  - Add cone glyphs to the stream tubes so you can see direction of the flow (see pages 35-36 of the ParaView Tutorial).

Do this with two types of seeds (i.e. cloud vs. line) and describe the different trends you can identity in the data with different seed types such as rotation of flow or how the air travels into and out of the center of the hurricane.

Cloud seed:

1) Wind flow streamlines can be seen in Figure 7.

2) Wind flow stream tubes can be seen in Figure 8.

3) Wind flow stream tubes with cone glyphs can be seen in Figure 9.

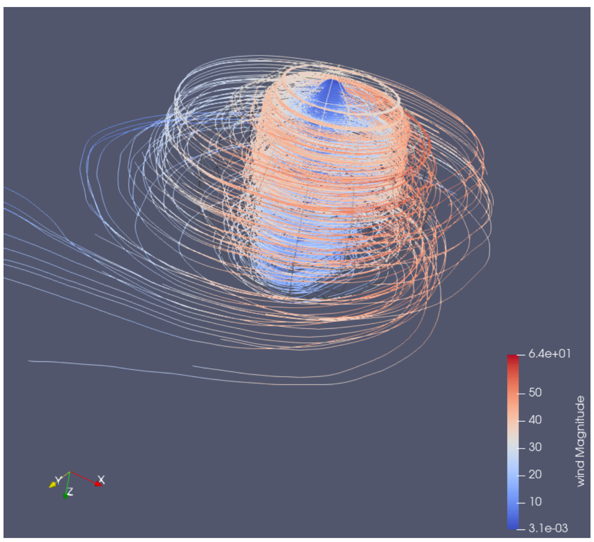

Figure 7: Part 2: streamlines (cloud seed)

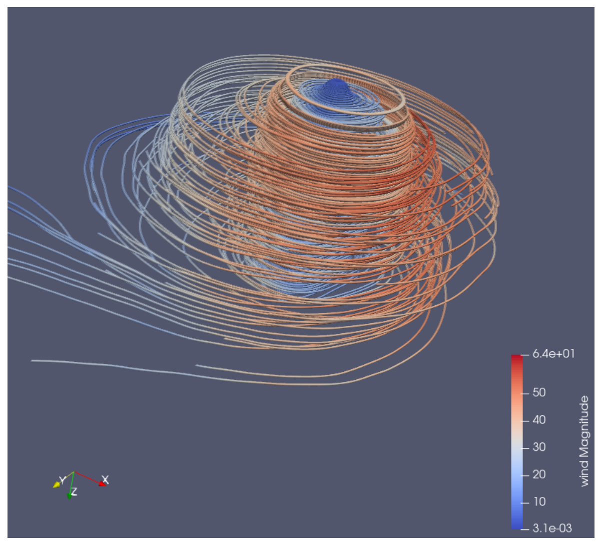

Figure 8: Part 2: stream tubes (cloud seed)

Point seed:

2) Wind flow stream tubes can be seen in Figure 10, I was not able to separate the lines from the tubes. Therefore, the first step could not be achieved as the visualization was created in tubes initially.

3) Wind flow stream tubes with cone glyphs can be seen in Figure 11.

Overall, we can tell that the wind currents go outwards from the ”eye” of the hurricane, and at the borders of the hurricane, some currents go in different directions, especially as seen on the point seed through cone glyphs. As seen on through the point seed representation, the hurricane forms at the bottom and evolves upwards and out.

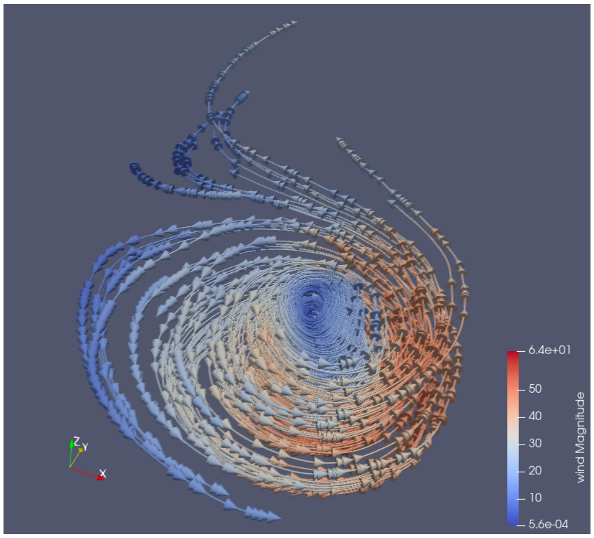

Figure 9: Part 2: stream tubes with cone glyphs (cloud seed)

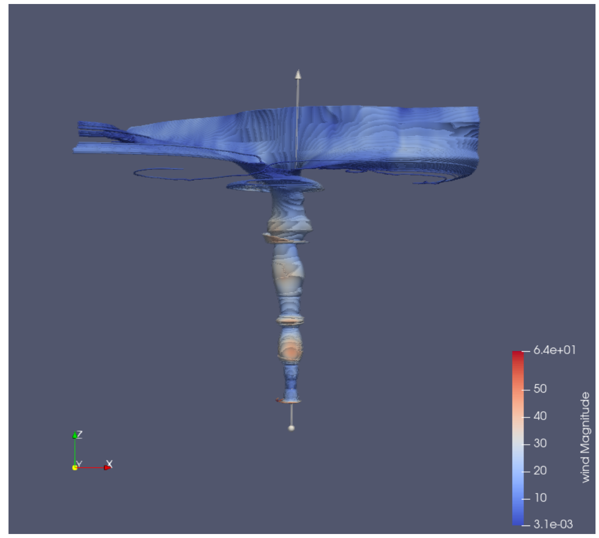

Figure 10: Part 2: streamlines (point seed)

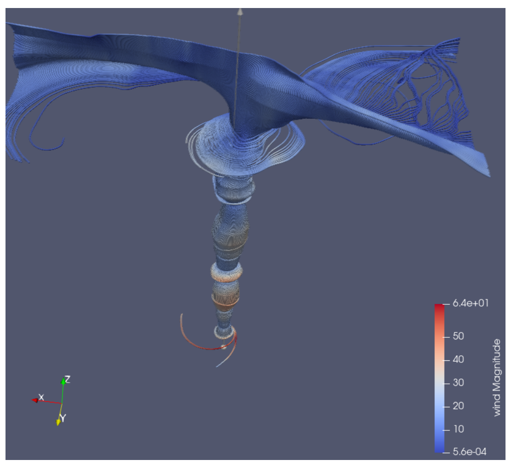

Figure 11: Part 2: stream tubes with cone glyphs (point seed)

- Step 2:
  - To see a big picture of the direction of the wind flow within the hurricane, create a visualization using arrow glyphs at randomly sampled places throughout the volume. You should use enough arrow glyphs to get a good overview, but not so many arrow glyphs that the view is cluttered.

    The result of the above instructions can be seen in Figure 12.
  - Now scale the vectors proportional to their speed and add a colormap for the speed. What happens, why should we do this? Explain the flow in this visualization and compare to any trends you observed in Step 1.

    The result of the above instructions can be seen in Figure 13.

    According to the new observations through randomly sampled glyphs, the hurricane speed is much higher at the top of the hurricane, and some of the currents go in different directions when the less dense the hurricane gets at the sides, which we could not observe in previous parts. It is

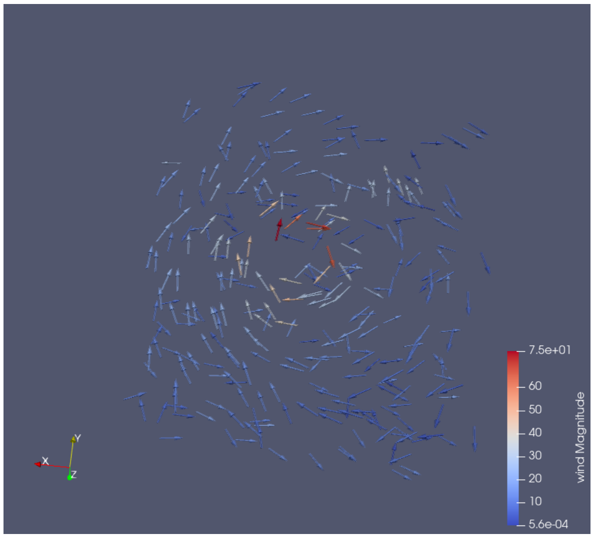

Figure 12: Part 2: visualization through randomly sampled glyphs

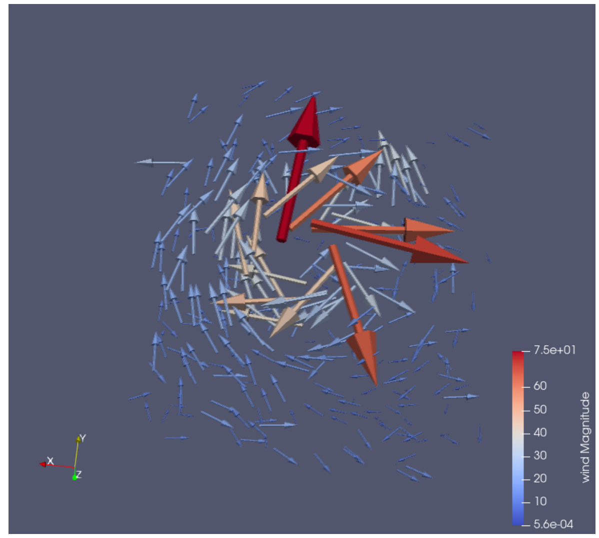

Figure 13: Part 2: visualization through randomly sampled glyphs scaled to vector speed

also noticeable that the currents at the bottom of the hurricane are flowing through the hurricane itself rather than following the usual circular motion.

3 Part 3: Visualization of Air Flow around a Moving Car

- Load the dataset: https://my.eng.utah.edu/ cs6635/Assignment4-output.vti.gz
- Design a transfer function for direct volume rendering of flow velocity magnitudes to create a visualization similar to the following (Red: relatively high magnitudes, Blue: relatively low magnitudes)

  The result of the instructions above can be seen in Figure 14.

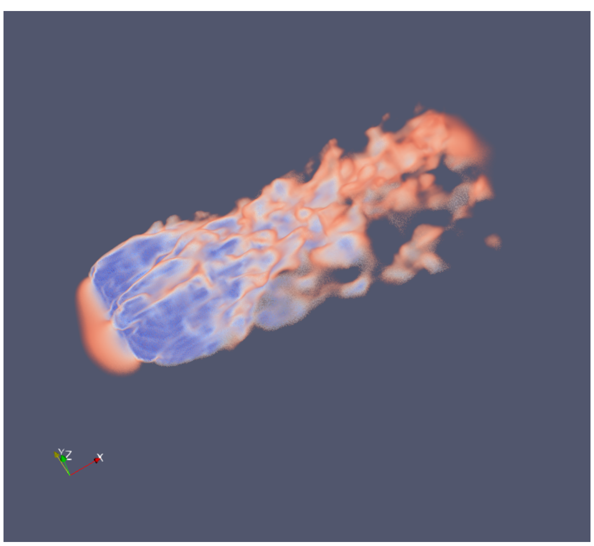

Figure 14: Part 3: direct volume rendering of flow velocity magnitudes

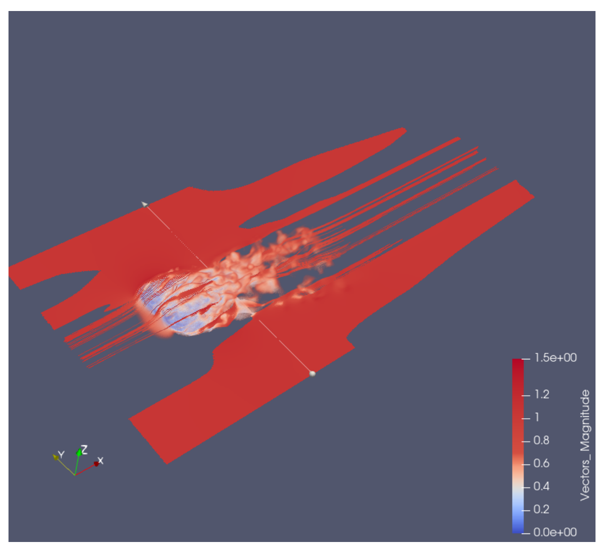

Figure 15: Part 3: streamlines (y axis parallel seed)

code: https://github.com/milenabel/CS6635-VisforScD

- Apply the Stream Tracer filter to output.vti. Please orient a seed line orthogonal to the moving direction and keep it horizontal before applying the filter.

  The result of the instructions above can be seen in Figures 15 and 16. Figure 16 represents the same seed with adjusted width.

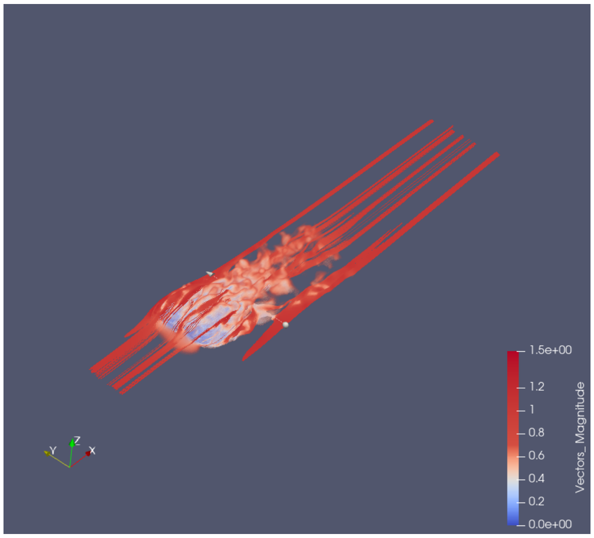

Figure 16: Part 3: streamlines with adjusted width (y axis parallel seed)

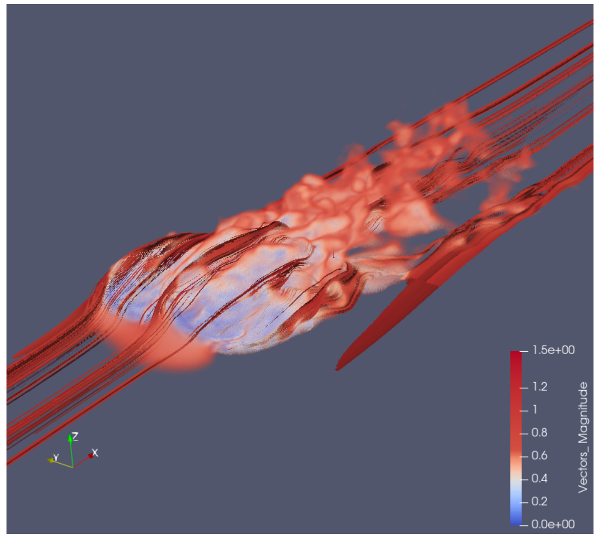

Figure 17: Part 3: streamlines as ribbons (y axis parallel seed)

- Apply the ribbon filter to stream lines. Set ribbon width parameter such that the resulting visualization has a pleasing appearance. Note the ribbons, in contrast to tubes, enable to better illustrate small-scale helicoidal trajectories (in turbulent areas).

  The result of the instructions above can be seen in Figure 17, the chosen width value is 0.15 here and from now on.

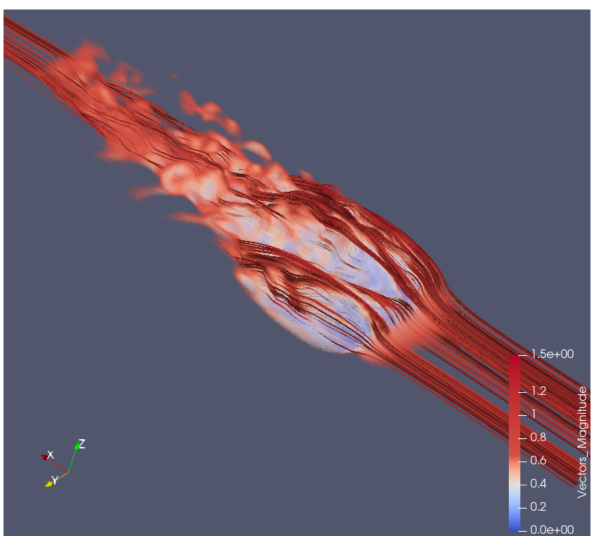

Figure 18: Part 3: streamlines as ribbons (x axis parallel seed)

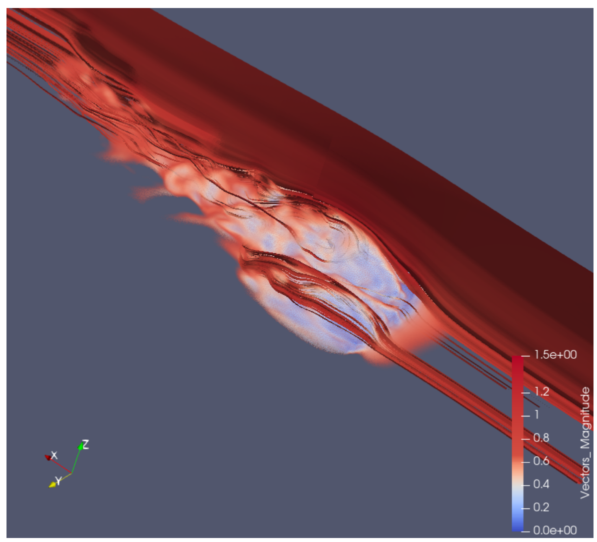

Figure 19: Part 3: streamlines as ribbons (z axis parallel seed)

code: https://github.com/milenabel/CS6635-VisforScD

- Create visualizations similar to image in question 4 for two more seed configurations of your choice. Please include screenshots with brief description, and a PSVM file.

  The result of the instructions above can be seen in Figures 17 and 18, I have chosen to use parallel to x and z axis seeds.

- Note any interesting observations for flow visualizations.

  We can notice that the flow of air around the car is consistent with the shape of the car and is more prominent on the sides where there are less barriers.

4 Part 4: Euler's Method

- Use random sampling to generate 15 seed points within the range [0,19] in both dimensions. Also, set the random seed manually to an arbitrary number. This will give you consistent seed points which will be helpful for comparison. Show an image of your plot.

  Initial plot of the data is seen in Figure 20, and Figure 21 shows the image of the plot with 15 seed points.

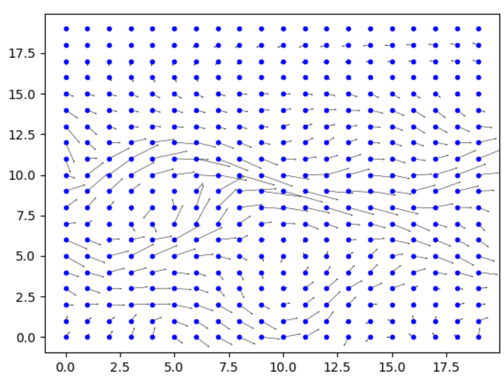

Figure 20: Part 4: initial plot of the data

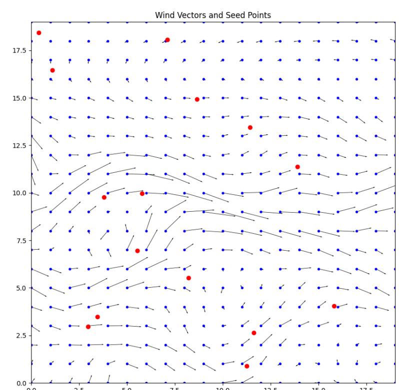

Figure 21: Part 4: plot with 15 seed points

- Trace a streamline from each point. Use a time step value of 0.3 and perform 8 steps for each streamline. You will need to write a function that will calculate the bilinear interpolation of the 4 neighboring vectors. Bound your points to the data range [0,20] along both dimensions. You can stop tracing early if you reach the image boundary. Show an image of your plot.

  The image of a stremline tracing plot with a time step value of 0.3 performing 8 steps for each streamline can be seen in Figure 22.

- Trace streamlines again from the same seed points as the previous part. Make 3 more figures with the following parameters:

  – Step size 0.15, steps 16

  – Step size 0.075, steps 32

  – Step size 0.0375, steps 64

code: https://github.com/milenabel/CS6635-VisforScD

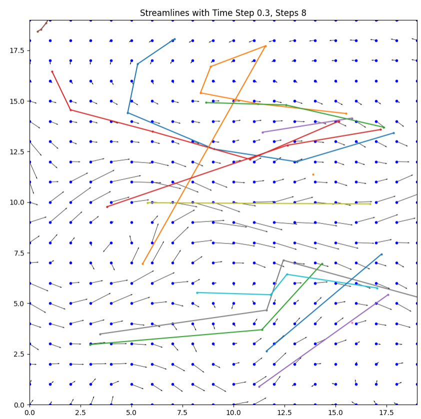

Figure 22: Part 4: streamlines tracing (step size 0.3, steps 8)

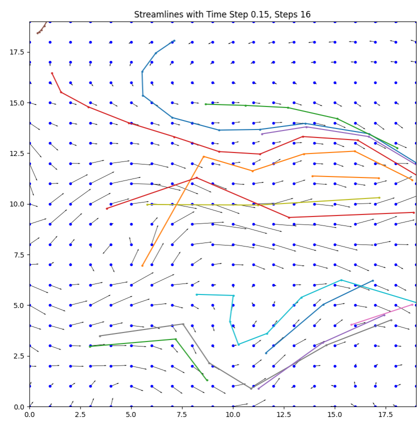

Figure 23: Part 4: streamlines tracing (step size 0.15, steps 16)

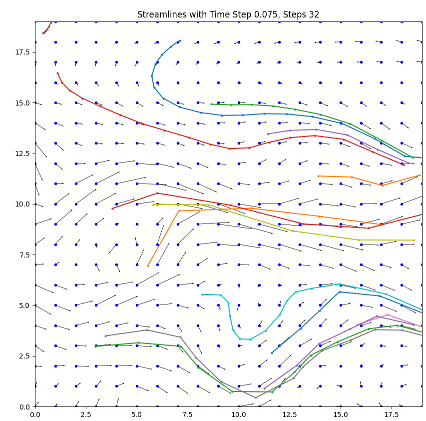

Figure 24: Part 4: streamlines tracing (step size 0.075, steps 32)

The images of streamline tracing plots with time step values of 0.15, 0.075, and 0.0375 performing 16, 32, and 64 steps for each streamline can be seen in Figures 23, 24, 25.

- Describe differences you see between the figures. Explain what divergence is. How does this relate to your results? (Tip: Read section 1.3.4 in the book)

  Analyzing the images for the streamlines generated using the Euler method, we observe the following differences as the step size decreases and the number of steps increases:

  – The streamlines appear smoother and more continuous with smaller step sizes and increased steps.

  – There is an increase in the accuracy of the flow representation, especially near complex flow features such as vortices or areas with abrupt changes in direction.

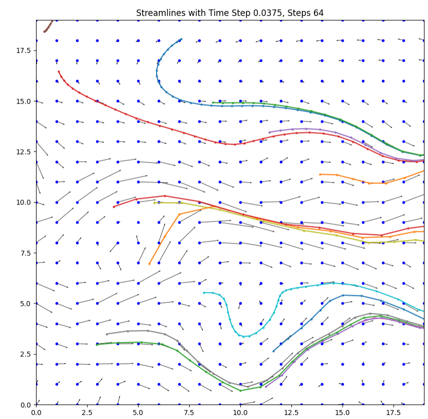

Figure 25: Part 4: streamlines tracing (step size 0.0375, steps 64)

- Streamlines in figures with smaller step sizes and more steps tend to follow the vector field more closely, revealing more detailed behavior of the flow.

Thus, the streamlines in each figure increasingly capture the local variations in the vector field, demonstrating the effect of the Euler method's accuracy on the visualization of the flow.

Divergence refers to the measure of the vector field's tendency to originate from or converge into a point. In the context of our results, divergence can be related to areas where streamlines are spreading apart (positive divergence, indicating a source of flow) or coming together (negative divergence, indicating a sink). In practical terms, if streamlines generated from closely situated seed points diverge rapidly, this indicates a region of high divergence.

Part 4.5 Extra Credit: Runge-Kutta Method

- Implement the Runge-Kutta Method(RK4) for the integration in part 4. Recreate the 4 figures from that question. When some of your sample points for interpolation are outside of the image bounds, only use the valid samples.

  The images of streamline tracing plots with time step values of 0.3, 0.15, 0.075, and 0.0375 performing 8, 16, 32, and 64 steps for each streamline generated using Runge-Kutta method (RK4) can be seen in Figures 10, 11, 12, and 13.

- Compare the results from RK4 with Euler.

  When comparing the streamlines generated by the Euler method with those generated by the Runge-Kutta 4th order (RK4) method, we can notice the following:

  - The RK4 method generally produces smoother streamlines than the Euler method for the same step size and number of steps.
  - RK4 results show a better approximation of the flow field with less numerical dispersion and fewer artifacts.
  - Differences between Euler and RK4 are more pronounced for larger step sizes.
  - As the step size decreases, the accuracy of the Euler method improves, and its results become closer to those of RK4, albeit still less accurate.

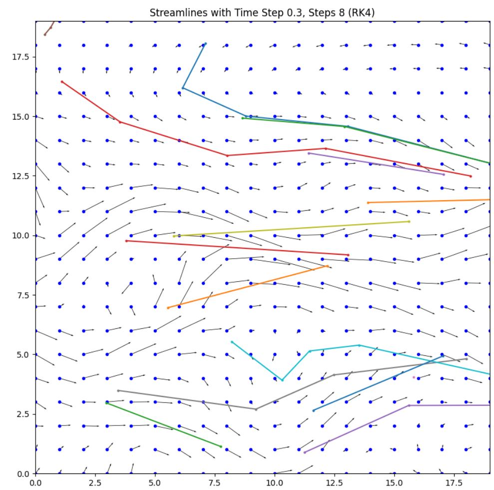

Figure 26: Part 4.5: RK4 streamlines tracing (step size 0.3, steps 8)

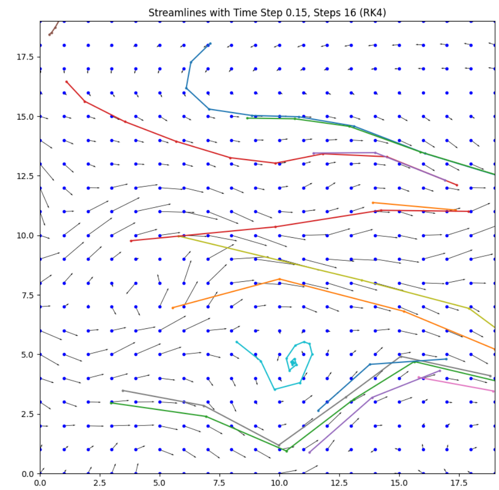

Figure 27: Part 4.5: RK4 streamlines tracing (step size 0.15, steps 16)

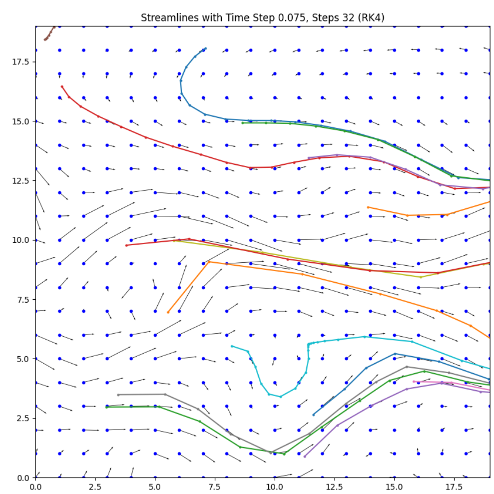

Figure 28: Part 4.5: RK4 streamlines tracing (step size 0.075, steps 32)

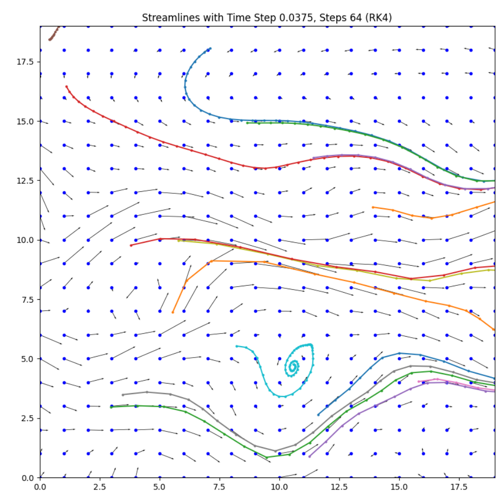

Figure 29: Part 4.5: RK4 streamlines tracing (step size 0.0375, steps 64)

- RK4 is less sensitive to the choice of step size in terms of accuracy, making it more robust for integrating the trajectories of particles in a vector field.

In summary, the RK4 method provides a more accurate and stable solution for tracing streamlines in a vector field, especially when larger step sizes are used or when high accuracy is required. The differences in the streamline plots highlight the superiority of RK4 over the Euler method in terms of accuracy and stability.

5 Part 5: The Reading Questions [Chapter 12, visualization handbook]

- What are steady and unsteady state flows? What are pathlines and streamlines?

  Steady state flows refer to flow fields where the velocity at any given point does not change over time. Conversely, unsteady state flows are those in which velocity at points in the field can vary with time. In the context of flow visualization, pathlines represent the trajectory that a single particle will follow over time in a flow field, marking its actual path. Streamlines, however, illustrate the direction of the flow at a single moment in time, mapping out the direction that particles would take if introduced into the flow at that instant. In steady flows, streamlines, pathlines, and streaklines (the line traced by a continuous release of particles) coincide, but in unsteady flows, they can differ significantly because the flow's direction and magnitude may change at each point over time.

- Briefly describe any three classifications of vector-field visualization techniques.

  Point-based direct visualization: This method relies on depicting the vector field at discrete points, often using glyphs like arrows to indicate direction and magnitude. These glyphs serve as local probes to the velocity field and can be designed to avoid pattern distraction and represent flow properties at a specific point.

  Particle tracing: Visualization is based on tracing particles through the vector field to generate characteristic curves such as pathlines or streamlines. These lines intuitively represent transport along the flow and can range from simple line representations to complex geometric structures like streamtubes in 3D spaces.

  Feature-based visualization: This technique focuses on identifying and visualizing significant flow features like vortices, critical points, or shock waves. It abstracts away less relevant information, allowing users to perceive key aspects of the flow's behavior.

- State any three features for feature-based vector field visualizations. Describe any two features in detail. Why is feature-based visualization important for vector field data?

  Topology-based Visualization: This approach seeks to represent only the critical aspects of a vector field by identifying its topological features such as critical points and separatrices.

  Vortex Identification: Vortex cores are central to understanding rotational behaviors in a flow, and several methods are developed for robustly identifying them.

  Quantitative Visualization: Goes beyond qualitative visual patterns to provide measurable values for flow properties such as vorticity or divergence.

  Detailed Explanation of Two Features:

  Topology-based Visualization: By focusing only on the critical points and the flow separations they induce, this approach simplifies complex vector fields into understandable diagrams. This enables users to infer the overall flow structure from these simplified representations. It's particularly effective in less turbulent flows where critical points are limited and distinct.

  Vortex Identification: Since vortices are often associated with significant physical phenomena like lift in aerodynamics, identifying them correctly is crucial. Various methods exist to measure the rotation and identify the vortex core, ranging from local indicators like velocity gradients to global ones that consider the geometry of particle paths.

  Importance of Feature-based Visualization: Feature-based visualization is important for vector field data as it enables users to focus on the most relevant aspects of complex flows, reducing cognitive load

code: https://github.com/milenabel/CS6635-VisforScD

and helping to extract meaningful patterns that can inform better decision-making or deeper insights into the physical phenomena being studied.

code: https://github.com/milenabel/CS6635-VisforScD

6 Conclusion

In this assignment, I embarked on a comprehensive journey through the visualization of flow dynamics, utilizing a variety of techniques and tools. The key highlights of this exploration included:

Streamline Visualization of Air Flow: Through the use of glyphs, filters, and color schemes, I was able to reveal the intricate patterns of air flow above a heated disk, showcasing the power of streamline visualization in capturing dynamic movements.

Hurricane Visualization: By employing stream tubes, cone glyphs, and different seeding strategies, I delved into the complex flow patterns within a hurricane. This section illuminated the diverse behaviors of air flow, from rotational dynamics to the outward and upward movement of wind currents.

Visualization of Air Flow around a Moving Car: Direct volume rendering and streamline techniques were applied to visualize air flow around a car. This part emphasized the interaction between the car's geometry and the air flow, highlighting the aerodynamic effects.

Euler's and Runge-Kutta Methods: The comparison between Euler's method and the Runge-Kutta method for tracing streamlines from seed points demonstrated the importance of accuracy and stability in numerical simulations of flow fields.

Theoretical Insights and Practical Challenges: The assignment also included a theoretical discussion on steady and unsteady state flows, pathlines, and streamlines, as well as classifications of vector-field visualization techniques. This theoretical grounding provided a deeper understanding of the flow phenomena being visualized.

The tasks completed in this assignment offered profound insights into the dynamics of fluid flow and the effectiveness of various visualization techniques. Each part of the assignment presented unique challenges, from choosing the right visualization strategy to implementing numerical methods for streamline tracing. These experiences have enriched my understanding of both the theoretical aspects of flow dynamics and the practical application of visualization tools in scientific research. Through this assignment, I gained valuable skills in data visualization, an appreciation for the complexity of fluid dynamics, and a deeper insight into the role of visualization in scientific discovery.

code: https://github.com/milenabel/CS6635-VisforScD