Contrail Particle Visualizer: An Aircraft Contrail Formation Simulation Display Tool

Roberto Paoli, PhD
Research Assistant Professor
Department of Mechanical and Industrial Engineering
University of Illinois at Chicago

G. Elisabeta Marai
Associate Professor of Computer Science
Department of Computer Science
University of Illinois at Chicago

Shyam Patel
Student
University of Illinois at Chicago

Abstract

With the current climate crisis, it is more important than ever that we consider aircraft contrail parameterization results in the conservation equations used in global atmospheric models. The simulation tool displays particles of contrail structure observed at the end of the jet and vortex regime in a web-view that utilizes D3.js and Three.js cross-browser JavaScript libraries for producing dynamic, interactive data visualizations and 3D models. The work was evaluated through a small case study. The tool successfully enables the presentation of contrail particle data.

I. Introduction

Contrails are ice crystals generated by water exhaust from aircraft engines. Their contributions to the Earth Radiation Budget—the balance between incoming energy from the sun and the outgoing longwave and reflected shortwave energy from the Earth—is relatively unexplored. In order to learn more about the physics of contrails and their impact on the environment, Roberto Paoli, a research assistant professor in mechanical and industrial engineering, and Liz Marai, computer science associate professor at the University of Illinois at Chicago, have teamed up to work on the project titled, "High-performance Computing and Data-driven Modeling of Aircraft Contrails."

The 3D nozzle geometry and initial contrail particles data from this project were used to build an interface that displays contrail formation emanating from the jet nozzle at various time steps. To conveniently examine the change in particles at each step, the tool accommodates an animation feature that, when enabled via a play button, runs through the available time steps in sequence at regular intervals without interruption. For future application, a force-directed graph is implemented that includes nodes that encompass simulations and enables the user to directly position the nodes via a click-and-drag operation in order to access available simulation data.

II. Methods

To display the 3D nozzle geometry, Dr. Paoli provided a SolidWorks part (SLDPRT) file that contains 3D computer-aided design (CAD) data which can be rendered using SolidWorks Visualize software. From there, the rendered nozzle geometry was exported into a Wavefront OBJ data definition file that represents the 3D geometry in human readable format and a companion Material Template Library (MTL) file that describes the geometry's surface shading properties. Both of these files are loaded into the web-view's Three.js WebGL renderer scene during run-time using the OBJLoader and MTLLoader loaders, respectively.

To display the contrail particles, Dr. Paoli's team provided particle data for seven time steps in a ParaView project. Using the reader module supplied by OpenFOAM's computational fluid dynamics

(CFD) software, the contrail formation was visualized for each of the provided time steps in the accompanying ParaView visualization application. From there, the x, y, and z particle positions were exported into individual comma-separated values (CSV) files. Using D3's built-in CSV parser, the particle positions recorded in these files are loaded into the web-view's Three.js scene dynamically as points, rendered by the WebGL renderer using the gl.POINTS type primitive.

To drive the selection, animation and display of time steps, a D3 slider that directs the addition and removal of particles to and from the scene is used. The slider axes are restricted to the subset of available steps for the given simulation, and the selection of any missing or unavailable steps is precluded. When the animation is activated using the accompanying play/pause button, the slider is advanced to the subsequent available step at regular intervals of 2500 milliseconds. To provide adequate time for loading of particle data and to avoid short intervals of vacuity, the change in particles on the scene is rendered after 500 ms. Aside from this, scene rendering is restricted solely to window resizing and user interaction with the model using the Three.js orbit controls (e.g., panning, zooming and rotating) to circumvent any unnecessary memory usage.

For future implementation, a D3 force-directed graph is added as a companion view that runs side-by-side to the simulation view. The nodes of the graph represent individual simulations and can be clicked, dragged and placed according to the user's preference. It is expected that, when additional data becomes available, the functionality of the graph will be extended to enable the selection and display of available simulation data.

The web-view of the tool can be accessed at https://mashy426.github.io, and particle data for the available steps is at https://github.com/mashy426/mashy426.github.io/tree/master/particles.

III. Evaluation and Results

To effectively evaluate the simulation tool, a small case study consisting of three users was conducted in two sessions throughout the semester. Feedback was used to improve upon the functionality of the interface. For instance, the animation feature was not initially implemented and it was unclear to users as to why certain time steps were not visible upon their selection. To address this issue and clear any ambiguity, multiple functions were added to drive and direct step selections, including the animated representation of the particles. Another problematic issue involved the web-view's responsiveness and memory usage. Due to the CPU-intensive nature of Three.js' perpetual rendering when the animation frame is requested, the optimal solution was to switch to rendering on demand. This involves manually rendering the scene whenever it is necessary, and not rendering when it is not.

Aside from these issues, the overall feedback received was positive. In addition to functionality, users appreciated the simplicity and ease-of-use of the interface. One user even commented that they had not encountered such an intuitive interface in the past. These results were highly encouraging.

IV. Discussion

The case study evaluation shows that the simulation tool is indeed adequate in presenting the available particle data through an effective and practical means. The tool is accurate in its display of contrail formation and user-friendly enough to be interpreted by users with minimal guidance. However, the testing could have been expanded to account for further analysis.

V. Conclusion

In conclusion, the particle visualization tool was introduced for displaying contrail formation emanating from the jet nozzle at various time steps. As more data by Dr. Paoli becomes available, the interface can be expanded to support its viewing.

Acknowledgments

Special acknowledgements to Dr. Paoli, his team, and Professor Marai at the University of Illinois at Chicago, without whom this project would not have been possible.