National Aeronautics and Space Administration

NASA Engineering and Safety Center Technical Bulletin No. 21-02

Genesis Flight Mechanics Simulation

The NASA Engineering and Safety Center (NESC) consolidated and modernized a suite of legacy flight mechanics simulations, including the Flight Analysis and Simulation Tool (FAST), resulting in Genesis, a generic, multi-vehicle, variable-degree-of-freedom flight mechanics simulation for ascent, aerocapture, entry, descent, and landing (A²EDL) trajectory design.

Genesis is more flexible, capable, and performant than FAST. It enables trajectory optimization and interactive trajectory generation. Its interoperability with Copernicus, an exo-atmospheric and interplanetary trajectory design tool, facilitates end-to-end trajectory optimization across all mission phases. Genesis is implemented in Julia, a new language for technical computing that combines the ease of use of scripting languages with the run-time performance of compiled languages.

Background

Flight mechanics simulations are used throughout a program's life cycle for tasks such as developing Design Reference Mission (DRM) trajectories for conceptual vehicle designs, evaluating or prototyping guidance algorithms, designing and reconstructing test flights, evaluating vehicle performance, and designing trajectories for operational missions. The JSC flight mechanics community relied on a suite of legacy flight mechanics simulations for these tasks. To simplify maintenance, improve on-boarding of new users, and better leverage modern high-performance computing (HPC) environments, the NESC consolidated the legacy simulations to create Genesis.

Benefits for the A²EDL Engineer

Julia is approachable to engineers without formal computer science training, which makes it easier for them to modify existing models or add new models. Julia does the heavy lifting to support multiple operating systems, including Windows, macOS, and Linux. Julia has a built-in package manager and a rich ecosystem of third-party packages, including optimizers, which can be used with Genesis. Julia has built-in support for linear algebra and Unicode variable names, which means that Julia code can closely resemble the textbook equation it implements.

Documentation: $\ddot{x} = u + R^T\left[\ddot{s} + 2\tilde{\omega}\dot{s} + \left(\dot{\tilde{\omega}} + \tilde{\omega}\tilde{\omega}\right)s\right]$

```
Julia: ẍ = u + R' * (s̈ + 2*ω̃*ṡ + (ω̇̃ + ω̃*ω̃)*s)
```

Figure 1: Julia code corresponds closely to the way an equation is documented.

Julia can be used in notebook programming environments that allow engineers to mix expository text, executable code blocks, and inline code outputs. This enables engineers to turn their Genesis analysis into an interactive document.

Genesis and Copernicus can be used in conjunction to enable end-to-end trajectory optimization. With the Copernicus plug-in capability, Genesis can generate segments of the overall trajectory. Copernicus can pass optimization variables to Genesis, allowing Copernicus to optimize the entire trajectory at once.

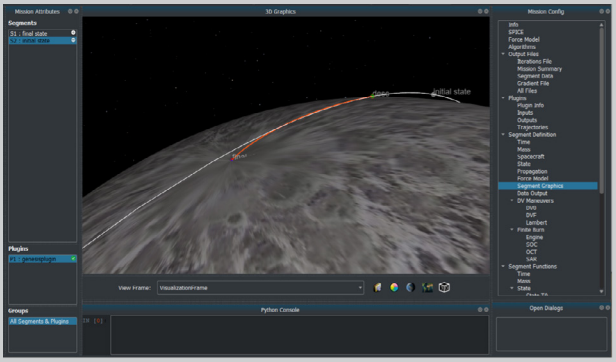


Figure 2: Copernicus and Genesis can be used together to enable end-to-end trajectory optimization. Here, the white trajectory is propagated by Copernicus, and the red trajectory is propagated by Genesis for a lunar descent and landing.

References

1. NASA/TM-20210014622, Improvements to the FAST and Initial Development of the Genesis Flight Mechanics Simulation for A²EDL Trajectory Design, April 2021
2. The Genesis GitLab page: https://gitlab-fsl.jsc.nasa.gov/Genesis/Genesis.jl (NASA-Only)
3. Julia main page: https://julialang.org

For information, contact Daniel G. Murri at daniel.g.murri@nasa.gov or Daniel A. Matz at daniel.a.matz@nasa.gov.