IAC-22,C1,IP,18,x73772

TudatPy - a versatile open-source astrodynamics software package for space education

Kevin Cowan[a]*, Filippo Oggioni[b], Geoffrey Garrett[b], Miguel Avillez[b], Joao Encarnacao[b], Marie Fayolle[b], Jonas Hener[b], Marc Naeije[b], Maarten van Nistelrooij[b], Ron Noomen[b], Michael Plumaris[b], Jeremie Gaffarel[b], Sean Cowan[b], Dominic Dirkx[b]

[a] *Space Department, Faculty of Aerospace Engineering, Delft University of Technology, Kluyverweg 1, 2629 HS, Delft, Netherlands* E-mail: `k.j.cowan@tudelft.nl`

[b] *Space Department, Faculty of Aerospace Engineering, Delft University of Technology, Kluyverweg 1, 2629 HS, Delft, Netherlands*

* Corresponding author

Abstract

The TU Delft Astrodynamics Toolbox (Tudat), is an open-source astrodynamics toolbox written in C++ with a focus on numerical state propagation. It has been under development by staff and students of the Astrodynamics and Space Missions (AS) section of Faculty of Aerospace Engineering at Delft University of Technology (TUD) for more than ten years. During the process, Tudat has become a key part of the Space Exploration MSc curriculum at TUD. Moreover, Tudat has been used in dozens of MSc projects, 5 PhD projects, and has contributed to about 20 peer-reviewed publications.
Tudat provides both numerical state propagation and state estimation functionalities, organized in a modular setup. The design of Tudat ensures mutual consistency between the various environmental and dynamical models. Further development of Tudat is currently underway or planned in several categories, such as the integration with Machine Learning frameworks, the development of Precise Orbit Determi- nation functionalities with real mission data, and the addition of more mission design and optimization techniques, including the integration with ESA's PaGMO/PyGMO toolbox.
Despite a continuous growth seen in the user base, the Tudat developer team has made the decision to create a Python interface for the software, motivated by the necessity of providing a more accessible and user-friendly tool that can be used not only by students and professionals, but also by space enthusiasts. Indeed, as a modern, high-level programming language, Python provides lower barriers to entry for users, signifcantly increasing the value of the Tudat kernel in education and academia. Named "TudatPy", this project has now become an ongoing effort to improve the accessibility of Tudat and to further develop the Application Programming Interface (API) for integration with external optimization and machine learning frameworks, while fully exploiting the Python language.
This paper thus demonstrates the use of the TudatPy API to 1) carry out an arbitrary optimization routine for a spacecraft mission using analytical methods, 2) perform the numerical simulation of the determined mission profile, 3) demonstrate a numerical analysis involving various integration schemes, acceleration models, and propagation methods, and 4) highlight various possibilities for uncertainty analysis and simulated trajectory estimation.
As an open-source project, TudatPy, and its installation and documentation, can be found at:
`https:tudat-space.readthedocs.io/en/latest/`

Guidance to tudat-team authors:
DO NOT EDIT THIS MAIN FILE. TO ADD YOUR MATERIAL, PLACE IT IN A SEPARATE .TEX FILE, WHICH WILL THEN BE IMPORTED USING AN `\INPUT{}` STATEMENT.

1. What is TudatPy?

Here's an example reference. [1]
[Target size of text in this section: ½ an A4]
[Input required here.]
What can you do with Tudat?
Consider the following:

- What are the high-level areas in which TudatPy can be applied?

- From the point of view of a user, what can you do with it?

The primary user, for the purposes of this paper, is assumed to be: someone interested in using TudatPy in a way similar to that of a TU Delft Space instructor, so for education which has a strong basis in and connection to research.

Guidance to tudat-team authors:
To do:

- ☐ As features are discussed, list published work (or educational settings) in which the various features have been applied.

[Input required here.]

2. Why use TudatPy?

[Target size of text in this section: ½ an A4]
[Input required here.]
Why should someone choose to use TudatPy? What are the main advantages of using TudatPy (eg. versus other packages → without naming them (?))?
Advantages / USPs to consider highlighting:

- Modularity
- Fidelity
- Usability, particularly with the Python interface as well as how this allows the use of the vast range of Python tools now available (eg. Numpy, SciPy, Matplotlib, scikit-learn, ...)
- Performance

Guidance to tudat-team authors:
To do:

- ☐ As features are discussed, list published work (or educational settings) in which the various features have been applied.

[Input required here.]

3. Examples of applications

The aim for this section, and thus for the case studies (aka examples), is to have all of the cases focused on a single mission/dynamic scenario. The proposed scenario is to focus on a mission from Earth to an asteroid such as Itokawa. NB: A mission to this asteroid is the topic of the example application using PyGMO in the documentation, which can be found here: docs.tudat.space

3.1 Analytical state propagation

[Target size of text in this section: one A4]
[Input required here.] Proposal:
Two-part presentation:

1. Single arc propagation
2. MGA-DSM

[Input required here.]

3.2 Numerical state propagation

[Target size of text in this section: one A4]
[Input required here.]

3.3 State estimation

[Target size of text in this section: one A4]
[Input required here.]

3.4 Trajectory Optimization

[Target size of text in this section: one A4]
[Input required here.]

Guidance to tudat-team authors:
Without extensively advertizing for PyGMO, mention the integration with PyGMO and the key advantages of integrating it into TudatPy, such as the open-source nature, successful track-record, ..., of PyGMO.

4. Project setup

[Target size of text in this section: ½ an A4]
Overview of:

- how the Tudat/TudatPy project is set up (eg. tudat vs. tudatpy, conda, github, build+CI pipeline, documentation)
- why the project is set up this way (ie. the logic justifying the current setup)

[Input required here.]

5. Code setup

[Target size of text in this section: ½ an A4]
Overview of:

- Key points guiding the philosophy of structure the code (in its current guise); eg. why the code was organized into the existing modules;

Fig. 1: Caption for a figure (here, an image).

- key points driving the design of the code (in its current guise).

[Input required here.]

6. Get started with Tudat

[Target size of text in this section: ½ an A4]

A brief indication of where to find it, how to get started "playing" with it, how to contribute to development, ...

Guidance to tudat-team authors:
To do:

- ☐ Our documentation is rather good, so we do not need to duplicate it extensively here. Instead, the intention is to point to where the reader can get started and briefly highlight how easy it is to do so.

This is text in a LaTeXminipage environment, which could be useful.

Acronyms/Abbreviations

[Use this section if necessary.]

Nomenclature

[Use this section if necessary.]

Acknowledgements

Appendix A (Title)

Appendix B (Title)

References

[1] D. Dirkx, E. Mooij, and B. C. Root. "Propagation and estimation of the dynamical behaviour of gravitationally interacting rigid bodies". In: *arXiv* 31 (2019).