Open Source Implementation of A Fast and Precise N-body Low-Thrust Propagator

Weichen Xiao*,[1] Wen Han Chiu,[1] and Chit Hong Yam[2]

[1]*Department of Physics, The Hong Kong University of Science and Technology, Hong Kong*
[2]*ispace inc., Japan*

Keywords: Numerical propagator, Low-thrust, High speed, High precision, n-body

Predicting the orbit of a spacecraft accurately is essential to the design and operation of all space missions, particularly for missions that involves multiple bodies and precise control, e.g. a cubesat mission to the Earth-Moon libration point which requires low-energy flybys at the Moon [1]. However to this date, there is no high precision propagator available to the research and industrial community as a benchmark. In this paper, we introduce an open source, highly customizable, fast and accurate N-body and low-thrust propagator which aims to fill in this gap.

Our numerical propagator is a essentially a program predicting the trajectory of a spacecraft with a specific low-thrust control in the gravitational field of multiple natural celestial bodies, with additional forces such as solar radiation pressure. The program is written in C++, which is a compiled language that runs much faster than interpreted languages. As an example, our program is compared to another numerical propagator written in MATLAB. Both programs are ran 100 times in a row with the same initial conditions in the same hardware and the same level of precision. For a trajectory of 1 year involving the sun and all the planets, it takes the MATLAB function about 60 seconds to finish the 100 runs, while our C++ program takes about 10 seconds according to the system timer, 6 times faster than the MATLAB program.

The integration method used in this program to solve the system of differential equations is the Runge-Kutta Cash Karp 5th order method with adjustable step size. This is provided by the C++ library of boost [2]-odeint [3], which is frequently updated and maintained by the open source community. In addition, the integration method, the precision and step size can be customized freely by the user. The states and mass of the natural celestial bodies are provided by the NAIF SPICE toolkit [4], together with ephemerides specified by the user. It is also up to the user to decide how many and which planets are going to be considered, and which ephemerides to use.

The user can specify the low-thrust control in either inertial frames such as ecliptic and equinox of J2000, rotating frames such as velocity-conormal-normal, and r-theta-h. The control profile can be specified as a function of time as a series of Chebyshev polynomials. The center of reference can be either in Solar System Barycenter (SSB) or non-inertial reference points such as the Earth, in which the acceleration of the Earth relative to SSB will be considered.

Here we present a test case using the actual data of the Cassini mission between June 1st, 2001 and May 22nd, 2002, a flight of 360 days during which the Cassini is on its way to Saturn after its flyby at Jupiter. The actual trajectory is retrieved from the JPL Horizon website [5], and the initial state on June 1st, 2001 is used as the input of both the C++ and the MATLAB program. Both programs are using the same absolute and relative tolerance of 10^{-14}, and including the Sun, all the planets, Pluto and the moon. A comparison of the the prediction of the C++ program and the MATLAB program is shown below.

TABLE I: Comparison of Integrating the Cassini Trajectory

Duration	Difference of X norm	Difference of V norm	C++ Runtime	Matlab Runtime
360 days	1.80×10^{-6} km (about 1.8mm)	2.31×10^{-13} km/s	About 0.1s	About 0.6s

The precision can also be checked by comparing the result of forward then backward integration to the initial state. We use here a trajectory with constant low-thrust in the inertial X-axis over a year.

TABLE II: Comparision of Integrating a Low-Thrust Trajectory

Duration	Difference of X norm	Difference of V norm
360 days forward and backward	1.34×10^{-5}km	5.29×10^{-13}km/s

To sum up, our program is fast and precise and it can be modified to accept different forms of inputs, natural celestial bodies, ephemerides, integration methods and other perturbating forces. We are willing to make it free and open source to share it among the community of astrodynamics academia and industry. We hope that it will become a standard benchmark software of high precision propagator in the future.

[1] Campagnola, S., Ozaki, N., Oguri, K., Verspieren, Q., Kakihara, K., Yanagida, K., Funase, R., Yam, C. H., Ferella, L., Yamaguchi, T., Kawakatsu, Y., Yarnoz, D. G., "Mission Analysis for EQUULEUS, JAXA's Earth-Moon Libration Orbit Cubesat," IAC-16-C1.5.10, International Astronautical Congress, Mexico, 2016.

[2] Boost C++ Libraries, http://www.boost.org/, retrieved on Nov. 7, 2016

[3] Odeint, http://headmyshoulder.github.io/odeint-v2/, retrieved on Nov. 7, 2016

[4] SPICE Toolkit - NAIF - NASA, https://naif.jpl.nasa.gov/naif/toolkit.html, retrieved on Nov. 7, 2016

[5] HORIZONS Web-Interface - JPL Solar System Dynamics - NASA, http://ssd.jpl.nasa.gov/horizons.cgi, retrieved on Nov. 7, 2016