VIPRE

Visualization of the Impact of the PRobe Entry location on mission and spacecraft design

A function index for the software package VIPRE, consisting of IPED and VAPRE, to compute (IPED) and visualize (VAPRE) the entry conditions for all safe entry opportunities of a planetary entry probe being released from an interplanetary trajectory.

It is a prototype and proof of concept on how to display all trajectory options and make the computed entry data available for filtering and browsing.

IPED

IPED – Matlab functions	
IPED_pythonInterface.m	Interface script called by Python setup function. Computes entry conditions for all safe entry opportunities for one trajectory.
setup_IPED.m	Script to compute entry conditions for all safe entry opportunities of a planetary entry probe being released from an interplanetary trajectory.
src/data	
Neptune.mat	Matlab formatted example trajectory data for Neptune.
Saturn.mat	Matlab formatted example trajectory data for Saturn.
Uranus.mat	Matlab formatted example trajectory data for Uranus.
src/utils	
arrow3.m	Script that draws lines from P1 to P2 with directional arrowheads.
circle.m	Rotates a circle with radius 'r', center coordinates 'o' from a plane with normal vector 'n0' = [0,0,1] into a plane with normal vector 'n'.
clearKernels.m	Clears previously loaded SPICE kernels for IPED-matlab computations.
closestApproach.m	Computes the point of closest approach of a trajectory with respect to the surface of a planet.

computingOrbStates.m	Computes orbital states based on one set of conical elements elts0 and the number of following steps n.
createPlots.m	Summarizes the code for creating plots with IPED. Same plots as in IPED_matlab. This function was initially called by IPED_python, however hasn't been used since after the initial testing phase.
Cross.m	Computes the vector cross product.
Ellipsoid.m	Calculates an ellipsoid given the ellipsoid semimajor axes 's', the unit vectors along the axes 'u', and the number of patches 'n' (similar to the sphere function from Matlab).
entryPoint.m	Defines the trajectory point at entry with an entry altitude 'hEntry' using 'idx' (altitude > hEntry) and 'idx+1' (altitude < hEntry).
eqtCrssng.m	Searches the trajectory state with the minimum distance to the equatorial plane for a given trajectory state set 'states' and its equivalent set of conical elements 'elts' by iterating and computing states between the two states that are enclosing the crossing.
hazardCrssng.m	Checks if the trajectory is passing through a hazard.
IPED_python.m	Main function for computations called by IPED_pythonInterface. Computes the entry conditions for all safe entry opportunities for one trajectory to the input 'body' with the identifier 'ID' and the hyperbolic velocity 'vInf', the arrival time 'epoch', at the entry altitude 'hEntry'.
IPED.m	Main function for computations called by setup_IPED.m. Computes entry conditions for all safe entry opportunities % for one trajectory to the input 'body'.
loadKernels.m	Loads necessary SPICE kernels for IPED-matlab computations. Takes the path to the generic SPICE folder as input.
Mag.m	Computes the magnitudes 'm' of each column of a given 3-by-n matrix 'u' with each column representing a vector.
planetData.m	Load the planetary data of the body specified in the input string 'body'.
planetRotation.m	Computes the rotational period of the planet 'omega' in rad/sec.
plot_planetSphere.m	Creates and ellipsoid in the planet's shape. Takes the number of facets 'nFacets' and the planet's dimensions in x,y,z in 'radii' as input.
R_unitVector.m	Returns the unit vector of the R-axis 'R_uv' of the B-plane for an incoming hyperbolic entry trajectory using the unit vector 'S_uv' and 'T_uv'.
radiusNonSphericalPlanet.m	Returns the planet radius 'R' and ellipticity 'f' at a geocentric latitude 'lat' for a non-spherical planet (oblate in z) with an equal equatorial radius in x and y direction, given with the vector 'radii'.

ringCrssng.m	Checks if input state 'state' lies within the radial distance of the rings of the planet, with the geometric information of the planet given in 'rings'. Assumes a circular ring formation (particles of rings on circular orbits).
rotationalSpeed.m	Computes the rotational speed of a planet, given its 'radii' in x,y,z directions and a rotational velocity 'omega' at a given latitude 'lat'.
RV2FPA.m	Computes the flight path angle 'phi' for a cartesian state given with its position vector 'r', its velocity vector 'v' and the gravitational parameter 'mu'.
T_unitVector.m	Returns the unit vector of the T-axis of the B-plane for an incoming hyperbolic entry trajectory.
trajData.m	Loads the trajectory data of the planet specified in the input string planet from a Matlab data file with format '.mat'.
Unit.m	Unitizes vectors by column.
VinfThe2B_plane.m	Computes the geometry of the B-plane using the vertex angle 'B_theta', the hyperbolic excess velocity 'vInf' and the gravitational parameter 'mu'.
tests/	
testCase_Saturn.m	Test case: Saturn. Computes entry conditions for an arrival at the planet Uranus using an interplanetary trajectory.
testCase_Uranus.m	Test case: Uranus. Computes entry conditions for an arrival at the planet Uranus using an interplanetary trajectory.

IPED – Python functions	
computation_parallel.py	Algorithms for parallel / multi-core computations.
pathDef.m	Defines the necessary directories and paths needed for the IPED computations. Adds the paths to the Matlab path.
setup_parallel.m	Setup to compute entry conditions for all safe entry opportunities of a planetary entry probe being released from an interplanetary trajectory. Setup for parallel/multi-core computations.
setup.py	Setup to compute entry conditions for all safe entry opportunities of a planetary entry probe being released from an interplanetary trajectory. Setup for single-core and local computations.
src/data	
Neptune.csv	Example trajectory data input as .csv file for Neptune.
Saturn.csv	Example trajectory data input as .csv file for Saturn.

Uranus.csv	Example trajectory data input as .csv file for Uranus.
src/utils	
dataProcessing.py	Collection of functions used by setup.py, setup_parallel.py and computations_parallel.py

VAPRE

VAPRE.py	A software tool to visualize the entry conditions for all safe entry opportunities of a planetary entry probe being released from an interplanetary trajectory. It is a prototype and proof of concept on how to display all trajectory options and make the computed entry data available for filtering and browsing.
src/	
callbacks.py	Collection of functions that handle callbacks from graphic display of VAPRE.py.
content.py	Collection of variables that define the content of the visualization containers that are defined in view.py.
dataLoading.py	Collection of functions that load the IPED data input files and make them available for processing.
dataProcessing.py	Collection of functions that process the data for visualization.
graphics.py	Collection of functions that create graphics and visualizations used by VAPRE.py.
view.py	Collection of functions that create software architecture of the visualization platform.
src/data	
Neptune.csv	Example trajectory data input as .csv file for Neptune.
Saturn.csv	Example trajectory data input as .csv file for Saturn.
Uranus.csv	Example trajectory data input as .csv file for Uranus.
src/utils	
dataFunctions.py	Collection of functions that handle data.