AAS 17-651

A TOOL FOR IDENTIFYING KEY GRAVITY-ASSIST TRAJECTORIES FROM BROAD SEARCH RESULTS

James W. Moore*, Kyle M. Hughes†, Alec J. Mudek‡, and James M. Longuski§

A tool is presented that identifies desirable trajectory candidates from among tens of thousands of gravity-assist trajectories. A broad trajectory search technique creates an exhaustive set of possible trajectories to a given planet. From this dataset, our tool reveals candidate trajectories with user-defined characteristics. Typical discriminating characteristics are launch V-infinity, time-of-flight, and delivered mass. Mission planners evaluate and plot interesting trajectories from within the tool. Our tool generates catalogs of selected trajectories for further evaluation with higher-fidelity trajectory solvers. This paper outlines the key features of the tool and gives examples of typical analyses.

INTRODUCTION

The JPL Satellite Tour Design Program[1–3] (STOUR; pronounced "ess-tour") is a patched-conic trajectory solver that has been instrumental in finding gravity-assist paths for a variety of exploration scenarios.[4–12] A key feature of STOUR is the capability to perform exhaustive searches for trajectory solutions along a given gravity-assist path. When multiple paths are considered, STOUR is capable of producing more trajectory solutions than can be efficiently evaluated by an analyst. The large dataset may require the trajectory planner to select a few trajectories for further evaluation based on a single design goal (e.g. time of flight or launch V_∞). The task of sorting through many trajectory solutions to identify attractive candidates in a single attribute can be tedious. Comparing trajectories across multiple attributes is even more difficult and time consuming.

We have developed a tool that provides new capability to perform comparative analyses of thousands of gravity-assist trajectories generated by STOUR. The tool allows mission planners to identify trajectory candidates that best meet multiple criteria on parameters extracted (or derived) from STOUR output such as launch V_∞, time-of-flight, and delivered mass. Mission planners may review key parameters and plot candidate trajectories. A catalog of selected trajectories can be created for further evaluation with a higher-fidelity trajectory solver. The tool has proven useful in developing catalogs of attractive trajectories to the Ice Giants over the coming decades.[13,14]

*Doctoral Candidate, School of Aeronautics and Astronautics, Purdue University, 701 W. Stadium Ave. West Lafayette, IN 47907-2045. Currently at Intuitive Machines, 3700 Bay Area Blvd Suite 100, Houston, TX 77058, AIAA Member.

†Doctoral Candidate, School of Aeronautics and Astronautics, Purdue University, 701 W. Stadium Ave. West Lafayette, IN 47907-2045. Currently at NASA Goddard Space Flight Center, Navigation and Mission Design Branch, Code 595, Greenbelt, MD 20771, AIAA Member.

‡Graduate Student, School of Aeronautics and Astronautics, Purdue University, 701 W. Stadium Ave. West Lafayette, IN 47907-2045.

§Professor, School of Aeronautics and Astronautics, Purdue University, 701 W. Stadium Ave. West Lafayette, IN 47907-2045, AAS Member, AIAA Associate Fellow.

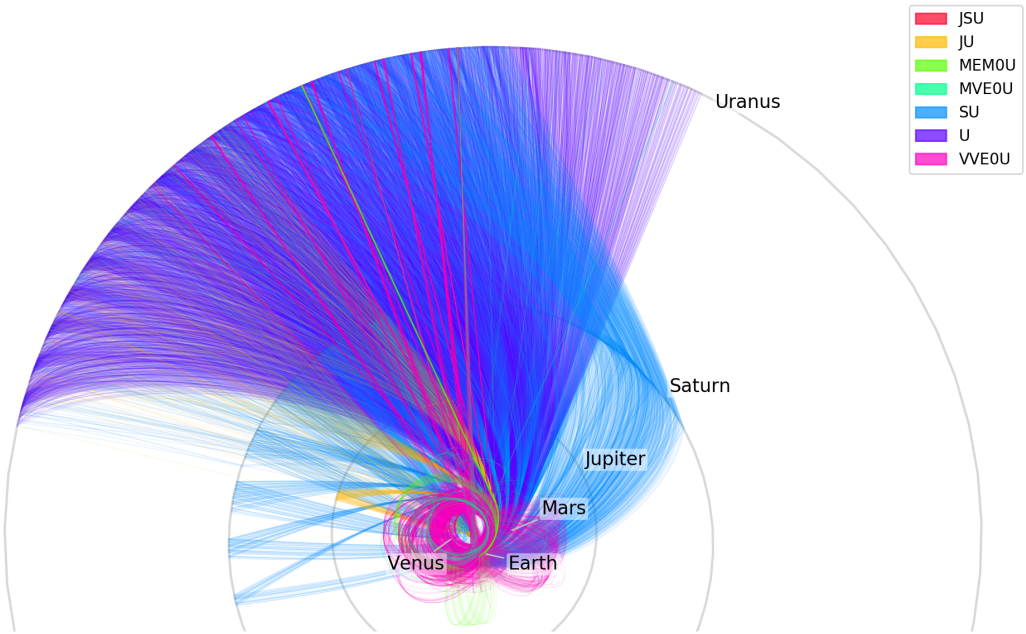


Figure 1. The figure above demonstrates the complexity of the broad trajectory search problem. The figure contains approximately 13,500 patched-conic trajectories to Uranus. The circular traces represent the orbits of the outer planets. Each color represents a different path. Each line represents a single solution along one of the paths with a specific launch date and energy. For example, the blue traces labeled "SU" for "Saturn-Uranus" are trajectories that leave Earth and fly by Saturn on the way to Uranus. Note: Mercury is not considered as a gravity-assist body in this dataset.

PROBLEM DESCRIPTION

STOUR has been successful at finding patched-conic solutions for multiple gravity-assist trajectories. Typically, researchers have employed STOUR to identify the solutions within a limited range of launch dates and launch V_∞ over a single flyby sequence (or "path"). More recently, we have been performing broad searches over decades of potential launch dates and over all feasible paths. These searches generate a great number of patched-conic solutions that require further study in higher-fidelity simulations. Figure 1 exemplifies the problem. Each color in Figure 1 represents a different path to Uranus. The paths are identified by the initials of the planets in the sequence with a zero representing a mid-course maneuver. Each line in the figure represents a unique conic solution with a specific launch time and energy. STOUR identifies all conic solutions that exist (subject to user-supplied constraints), even though they may be impractical for certain mission realities. Figure 1 presents only a few of the dozens of feasible paths that may exist. Because a higher-fidelity analysis on each solution is time consuming, we wish to limit the number of trajectories to those that are most attractive.

```
EVENT  1      2.   0, 1   0, 4
EVENT  2      3.   0, 1   0, 2
EVENT  3     OPTIMIZED MANEUVER
EVENT  4      7.   0, 0   0, 0
PATH:  3  2  3  0  7
LAUNCH DATES SEARCHED: 20230101 TO 20381231 BY  5.0 DAYS
VINF :  2.50 TO  12.00 BY   0.50 km/s
TFMAX =  5479.0 DAYS
ALTMIN =    300.KM
                                    SATELLITE  HP(KM)  THETA(DEG) VINF(KM/S)  PERIOD PERIAPSIS APOAPSIS   C3      TF   TOTAL TF
 EVENT ************ ISTR ************** PLANET   or VOUT or DV(KM/S)  or TOT DV  (DAYS) (AU/CBODY RADIUS)(KM^2/S^2)(DAYS)  (DAYS)
                                                                                  or -gload                  or -gload

 LAUNCH DATE = 01/06/ 2023 AT 00:00:00  LAUNCH V-INFINITY =  11.00  stour.bsp ibod = -1800000001  stour.flybys.bsp ibod = -1810000001

  1     2     1      2      2            Earth        200.0  -27.3115  11.0000    378.027   0.671   1.376           665.44   665.44
  2     3     0      1      1            Venus       4029.9  -24.5151   9.0341    621.698   0.713   2.138           577.90  1243.35
  3     0     1    200.  180.0  0    3.0 Earth        200.0  180.0000  14.7087   2264.290   0.886   5.864             3.00  1246.35
  4     7     0      0      1           MANEUVER              2.6152   2.6152   16273.32   0.883  24.253          3523.12  4769.47
  5                                     Uranus                        6.7713

 LAUNCH DATE = 01/06/ 2023 AT 00:00:00  LAUNCH V-INFINITY =  11.50  stour.bsp ibod = -1800000002  stour.flybys.bsp ibod = -1810000002

  1     2     1      2      2            Earth        200.0  -25.9804  11.5000    376.848   0.657   1.385           661.52   661.52
  2     3     0      1      1            Venus       3520.5  -32.5803   9.9208    621.645   0.701   2.150           580.92  1242.44
  3     0     1    200.  180.0  0    3.0 Earth        200.0  180.0000  15.0496   2228.755   0.875   5.803             3.00  1245.44
  4     7     0      0      1           MANEUVER              2.5407   2.5407   15550.22   0.876  23.509          3596.48  4841.92
  5                                     Uranus                        6.6139

 LAUNCH DATE = 01/06/ 2023 AT 00:00:00  LAUNCH V-INFINITY =  12.00  stour.bsp ibod = -1800000003  stour.flybys.bsp ibod = -1810000003

  1     2     1      2      2            Earth        200.0  -24.1499  12.0000    375.797   0.645   1.394           658.37   658.37
  2     3     0      1      1            Venus       2352.5  -40.5428  10.7402    621.391   0.689   2.161           582.94  1241.32
  3     0     1    200.  180.0  0    3.0 Earth        200.0  180.0000  15.3834   2189.984   0.864   5.737             3.00  1244.32
  4     7     0      0      1           MANEUVER              2.5136   2.5136   15004.12   0.868  22.942          3660.40  4904.71
  5                                     Uranus                        6.4865

 LAUNCH DATE = 01/11/ 2023 AT 00:00:00  LAUNCH V-INFINITY =  10.50  stour.bsp ibod = -1800000004  stour.flybys.bsp ibod = -1810000004

  1     2     1      2      2            Earth        200.0  -30.3039  10.5000    378.277   0.684   1.363           667.77   667.77
  2     3     0      1      1            Venus       2948.8  -17.5232   8.2082    622.811   0.723   2.131           574.75  1242.52
  3     0     1    200.  180.0  0    3.0 Earth        200.0  180.0000  14.4146   2315.470   0.895   5.955             3.00  1245.52
  4     7     0      0      1           MANEUVER              2.3462   2.3462   14372.36   0.899  22.238          3761.15  5006.67
  5                                     Uranus                        6.2763
```

Figure 2. Representative output from STOUR for a single gravity-assist path (Earth-Venus-Earth-Uranus with a maneuver after the second Earth flyby). Each table (starting with 'LAUNCH DATE') summarizes a different trajectory solution for the examined path. The boxes highlight features of the solutions that an analyst might weigh simultaneously. There may be hundreds of tables within a single output file. The tabular and chronological format does not lend itself to comparison of the various trajectories within the output file or to paths described in other files.

STOUR is quite capable of performing the trajectory searches, but the traditional output and post-processing tools are not well-suited for identifying the "best" trajectories among the results. Figure 2 presents a sample of an STOUR output file. Within the file are a series of tables summarizing the planetary encounters for each trajectory solution discovered along a given gravity-assist path.

For example, the line (highlighted in red) that reads, "LAUNCH V-INFINITY = 11.00" begins a table describing a patched-conic solution to Uranus that leaves Earth on January 6, 2023. The subsequent rows summarize the encounters at Earth, Venus, and a second pass of Earth. The next row provides information on an impulsive ΔV of $2.6\,\mathrm{km/s}$ that is the final trajectory event before arrival at Uranus with a V_∞ of approximately $6.8\,\mathrm{km/s}$. The time-of-flight for each leg and cumulative time-of-flight can be read from the rightmost columns. Our example trajectory requires $4769\,\mathrm{days}$. Several of the columns represent different parameters depending on whether the row describes a flyby or a maneuver.

The mission designer must weigh several competing parameters when selecting an appropriate trajectory. The designer may seek a trajectory with a short time-of-flight and a low launch V_∞, but these two parameters are likely to be at odds with each other. STOUR can be used to find preliminary values for many such characteristics, some of which (launch V_∞, propulsive ΔV, arrival V_∞, and time-of-flight) are highlighted in Figure 2. STOUR does the work of searching for potential flyby combinations and calculating the trajectory parameters. Our post-STOUR toolkit gives the mission designer new power to find the most attractive results.

The results shown in Figure 2 are only the first few solutions in a file that may contain thousands of solutions (depending on the granularity of the search and the chosen gravity-assist path). Finding the "best" trajectories along a given path is a multi-dimensional optimization problem with hundreds to thousands of trajectories to consider. Moreover, all the tables included an output file like Figure 2 are for a single sequence of flyby bodies. The problem grows more complicated when additional paths to the target planet are considered as shown in Figure 1. The final product should be capable of managing thousands of trajectories and be versatile enough to work for a variety of cost functions.

METHOD OF SOLUTION

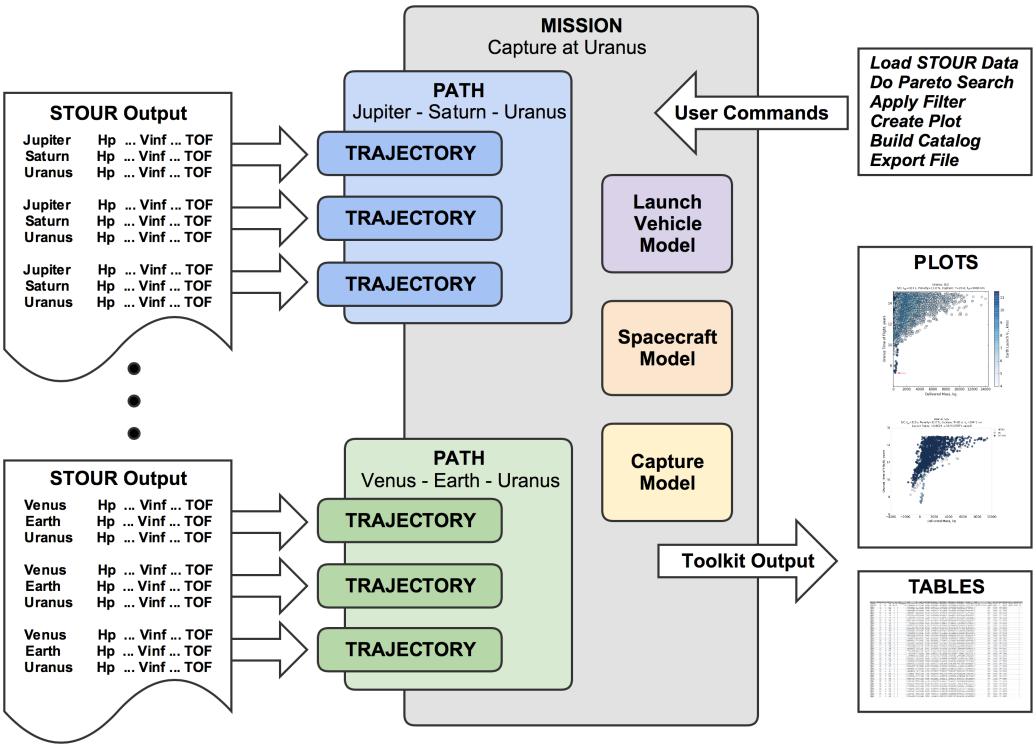


Figure 3. The toolkit models the possible trajectories as a series of nested objects. A Mission object contains multiple Path objects, each of which reach the target planet through a different series of gravity assists. The Path object contains multiple Trajectory objects, each of which is a patched-conic solution with different timing and energy. Legacy STOUR output (as shown in Figure 2) is translated and merged into the Trajectory and Path objects. The user provides input to the Mission object to define mission parameters, command actions, and retrieve output.

Our solution retains the pedigree and "speediness" of the compiled-Fortran STOUR heritage code and applies a set of generalized post-processing functions, written in Python[15*], to improve the user's ability to interact with large datasets in the traditional output format. The post-processor toolkit employs the Python pandas[16†] library to manage the data. We introduce a set of Python objects to represent gravity-assist concepts. The object methods simplify the interaction with the pandas database. Researchers typically arrange these methods in a script organized to perform standard analyses but the Python command line interpreter also provides a versatile interface.

*https://www.python.org
†http://pandas.pydata.org

We have applied an object-oriented approach to the toolkit development. In this paradigm, code is organized into a set of "objects" which contain data as "attributes". The objects typically have procedures or "methods" to act on the data. In a sense, objects are *nouns* and their methods are *verbs*. Objects are often abstractions of physical things but in our case they model astrodynamical concepts.

Figure 3 describes our toolkit software architecture. The toolkit parses the legacy STOUR text output (as shown in Figure 2) and populates a series of hierarchical objects with the extracted data. These objects provide an intuitive method of navigation through the STOUR results (from generic mission to specific trajectory). The user primarily interacts with the Mission object which represents the collection of possible trajectories to achieve a basic mission goal (i.e. arrive at a specified planet within an allotted time). The Mission object also contains very simple launch vehicle and spacecraft models that are user-configurable. Through the Mission object, the mission designer provides user input in the form of commands to import STOUR output, perform filtering, generate plots, and export results to tabular form. The Mission object contains a group of Path objects which represent different paths from Earth to the target planet by a specific series of gravity assists. Within the Path object is a group of Trajectory objects which represent actual patched-conic solutions along the specified path with various launch times, launch energies, and flyby times. The individual STOUR output text files translate into Path objects with each table in the file (see Figure 2) translating into a Trajectory object. The grouping of many paths into a common data object is a key innovation that enables new and broader mission design studies.

The Mission object includes a pandas DataFrame that assembles all the trajectory data from all the paths into a common location. The DataFrame enables dynamic sorting, indexing, and filtering of the data. The Mission object provides a wrapper around the DataFrame to simplify the data manipulation and plotting capabilities by organizing the required commands into object methods. Together, the methods allow the analyst to quickly produce standard analyses. The toolkit accepts optional inputs to the standard methods to provide a degree of flexibility. Direct access to the DataFrame allows complete versatility and insight into the data.

The toolkit is adept at filtering a large set of STOUR trajectory path solutions into a more manageable subset that meets certain design criteria of interest to the analyst. The plotting features include a graphical display of each trajectory in up to four user-defined dimensions for visual comparison. Built-in and user-defined filters also provide a means to discriminate between candidate trajectories. Output methods organize selected candidates into a variety of formats for record keeping or transfer to high-fidelity solvers.

The "User Commands" and "Toolkit Output" arrows in Figure 3 describe the interactive nature of the toolkit. A typical user-interaction session begins by initializing a Mission object. We may also configure a launch vehicle model, a spacecraft model, and capture orbit parameters (if these are needed for the analysis). We then instruct the Mission object to load a set of STOUR output files, which populates the Path and Trajectory objects. At this point, we can manipulate the dataset to conduct the analysis. For example, we may instruct the Mission object to perform calculations using the STOUR solutions and the spacecraft model, or apply some filtering to the dataset. We can request plots of any parameter vs. any other parameter and we can export tabulated data. We may also navigate down through the individual Path and Trajectory objects to further inspect any single STOUR solution. Finally, all of this interaction may be saved as a script so it can be re-used.

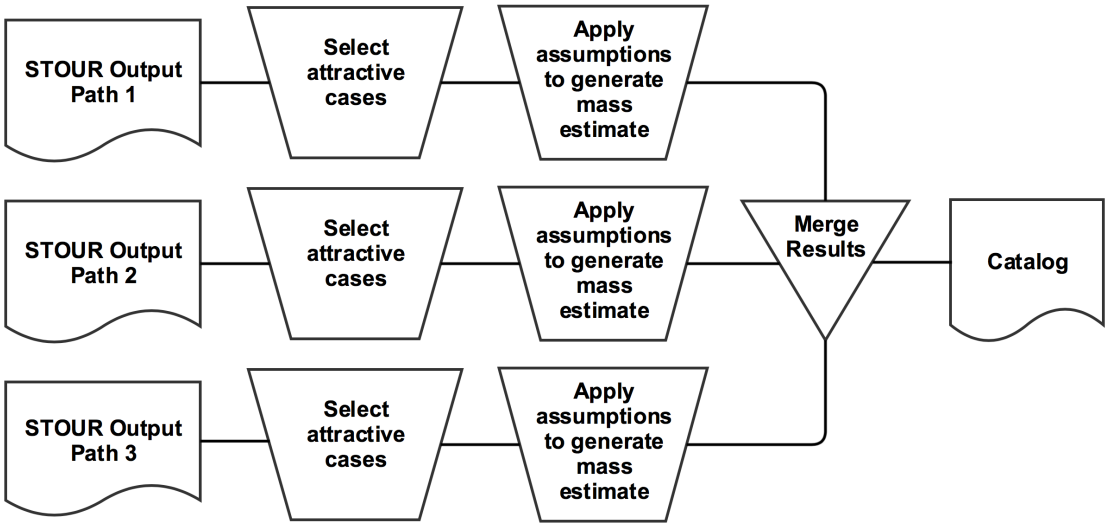


Figure 4. A traditional STOUR analysis work flow includes manual reduction of trajectories to be further analyzed. Multiple paths require a parallel process with additional manual analysis. The trapezoids represent labor-intesive processes that are performed on a limited number of trajectories.

FEATURES

Some key features of the tool are described below. The toolkit is highly configurable and easily adaptable to any study based on STOUR output. For demonstration purposes, the examples below assume the task is to produce a catalog of trajectories to an outer planet, considering multiple potential flyby configurations.

Multiple Path Comparison

Mission planners traditionally use STOUR to find patched-conic solutions in the ephemeris model along a specific flyby combination, or "path". The path is the ordered sequence of flyby bodies (e.g. Mars-Venus-Jupiter-Neptune). Path-solving is task of finding the launch dates, flyby dates, and velocities for which a path is viable. STOUR can solve for all the trajectories (subject to user-defined constraints) that follow a given path. Comparing the solutions for one path against an alternate path (e.g. Mars-Venus-Saturn-Neptune) is not straightforward since solutions for each path reside in a separate text file. Our tool combines the results of multiple STOUR studies into a single database to enable comparative analysis of dozens of potential paths to any given planet.

A traditional analysis work flow is pictured in Figure 4. In this example, the goal is to generate a catalog of desirable trajectories to a given planet by the following procedure:

1. Run STOUR for each potential path
2. For each Run:
   (a) Select the most attractive candidates based on an STOUR output parameter (e.g. launch date, time-of-flight, arrival V_∞, total ΔV)
   (b) Apply assumptions about mission parameters to convert STOUR output to mission design metrics
3. Assemble the best cases from each path into a catalog

The selection of the most attractive candidates typically involves weighing three or more parameters. The STOUR architecture is designed to walk through a range of launch times and launch energies to identify all the solutions that follow a single sequence of planets. As a result, the output (see Figure 2) is necessarily grouped into a single file for each flyby path. The solutions are ordered by launch date and launch V_∞. As discussed above, this legacy output format does not lend itself to comparison of the different solutions against one another. The number of trajectory solutions within a file can be significant. Previous tools (such as plots like Figure 6) enabled comparison of the many trajectories within a single STOUR output file to allow selection of attractive candidates even while considering multiple parameters. The direct comparison of multiple paths is solved by our new tool.

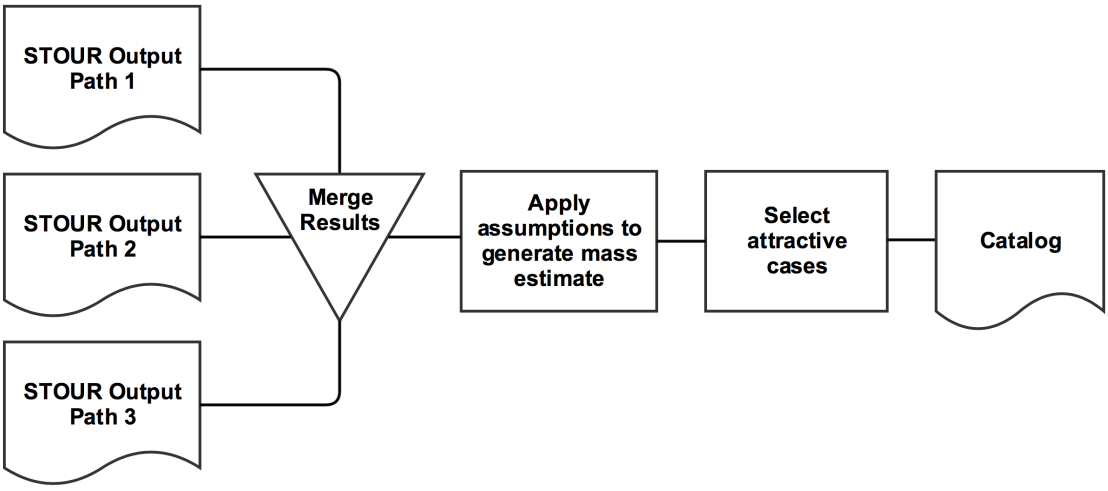


Figure 5. An example of a workflow with the toolkit is shown above. Labor-intensive processes have been automated and can be performed on all trajectories. Selection of desirable candidate trajectories has been moved to the final stage.

The toolkit enables simultaneous comparison of many trajectories from multiple paths by ingesting the legacy STOUR text output and compiling it into a single Python/pandas DataFrame. This workflow is described by Figure 5. The merging of the various path data takes place early in the process. The number of the repetitive tasks is reduced. The new workflow reverses steps 2(a) and (b) in the procedure above. With the STOUR output available as a Python data object, many of the previously labor-intensive processes have been automated. In particular, the steps that are required to convert the STOUR output into mission design metrics such as payload mass have been standardized and simplified. Deriving design metrics from STOUR output is discussed in the next section. The new procedure allows the analyst to compare multiple flyby paths simultaneously and directly in terms of the desired metrics.

STOUR Derived Data

A key feature of our architecture is the ability to update the database with information that can be derived from the original STOUR output. For example, the analyst may add a mass timeline to the database by choosing from a list of common launch vehicles, defining flyby or capture parameters, and providing notional propulsion characteristics of the spacecraft. Our tool uses this performance information and the STOUR propulsive ΔV output to add the spacecraft mass history as an attribute of each trajectory. The analyst may then use arrival mass to filter trajectories for further study.

Analysts can add and employ any characteristic that can be derived from standard STOUR output in a similar manner.

A traditional evaluation of STOUR results (for a capture mission) would compare all the solutions for a given path on a plot with arrival V_∞ on one axis and time-of-flight on the other. Additional information could be simultaneously evaluated by coding the plot point to represent launch V_∞. An example of this technique is shown in Figure 6 taken from Spreen et al.[10]

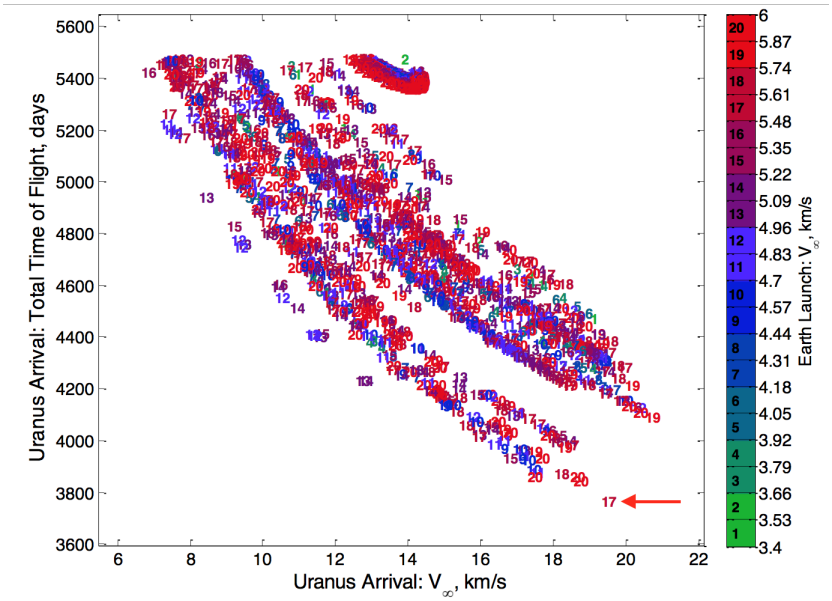
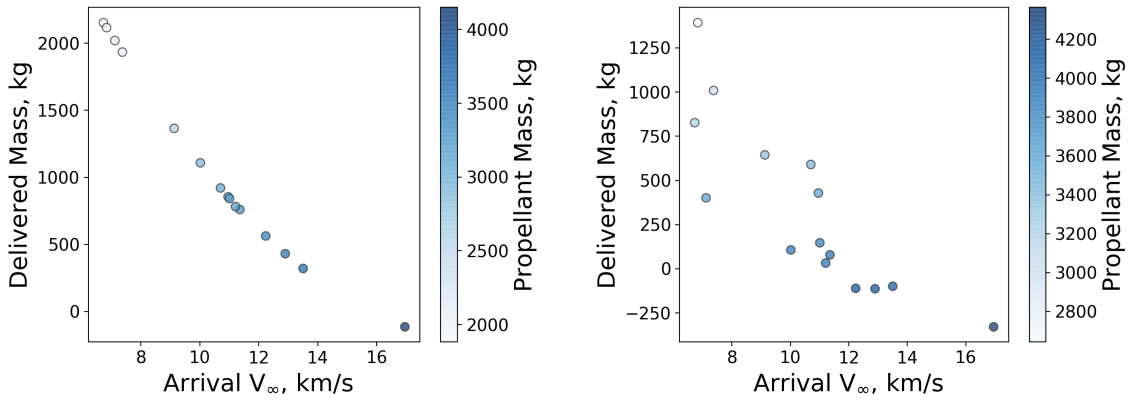


Figure 6. The Uranus trajectories from STOUR are plotted in this figure taken from Spreen et al.[10] In this type of figure, the numerical plot point (or its color) is a key for reading the launch V_∞ on the scale to the right. The arrival V_∞ and time-of-flight can be read from the coordinates of the point. For example, the point annotated by the arrow has an arrival V_∞ of 19.5 km/s, a total time-of-flight of 3800 days, and a launch V_∞ between 5.48 and 5.61 km/s. Note: figures like this are limited to a single flyby path, in this case VVEEJU (Venus-Venus-Earth-Earth-Jupiter-Uranus).

In the evaluation described above, analysts discriminate using V_∞ because it is a rough stand-in for payload mass. All other things being equal, the higher the arrival V_∞, the more mass will be required for capture at the target planet, and the lower the payload mass. However, all other things are not equal when we configure STOUR to provide mid-course maneuvers to close some borderline ballistic solutions. An additional evaluation step is required to determine if some attractive cases from a figure such as Figure 6 actually require a significant mid-course ΔV. An example of this problem is shown in Figure 7.

Figure 7 (a) shows the relationship between delivered mass when there are no mid-course maneuvers (or the propellant needed for those maneuvers is ignored). Under these assumptions, it is reasonable to select trajectories based on arrival V_∞ because a lower arrival V_∞ corresponds to a higher delivered mass. However, if mid-course maneuvers are present, as in Figure 7 (b), some trajectories with lower arrival V_∞ may actually deliver less mass than the higher V_∞ trajectories. In addition, the propellant mass may be great enough in some cases that the arrival mass becomes insignificant and the trajectory must be discarded.



(a) Without Maneuvers

(b) With Maneuvers

Figure 7. The plots above show the relationship between arrival V_∞ and delivered payload mass under two assumptions on an identical set of trajectories. In Figure (a), the propulsion system mass for the mid-course maneuvers is not taken into account. In this oversimplified approach, a lower arrival V_∞ always equates to a higher delivered mass. In Figure (b), the propulsion system mass required to perform the maneuvers is deducted from the arrival mass. In this approach, a lower arrival V_∞ does not always equate to a higher delivered mass. This nuance is not discernible from raw STOUR output.

The problem described above is important to consider when we wish to compare multiple path solutions, some of which may include mid-course maneuvers and others of which may not. STOUR does not consider spacecraft mass when solving for the patched-conic solutions. This is the reason for the mass estimation steps in Figures 4 and 5. The toolkit allows us to derive a mass history from the STOUR output and make comparisons directly in terms of mass delivered to the target planet.

Creating a mass history starts with an initial mass estimate. This is the mass to be assumed at Earth departure — the start of the STOUR search. Launch vehicles have established performance curves describing the mass that can be delivered to a given C_3 (or V_∞^2). Some examples of these curves are shown in Figure 8. The toolkit contains roughly 30 launch vehicle performance curves including some variants with upper stages. STOUR provides the launch V_∞ for each trajectory solution. This V_∞ is converted to C_3 and the performance curve for the user-selected launch vehicle provides the mass that can be delivered to the given V_∞.

Analysts provide the toolkit with gross estimates of spacecraft performance to complete the mass-based comparisons. We specifiy an I_{sp} value for chemical-propulsion maneuvers and use the rocket equation to determine the mass of propellant required for each maneuver in the STOUR solution. We specify a tanking penalty as a percentage of propellant mass to account for non-payload, structural mass that must be included in the spacecraft. If capture and orbit about the target planet is envisioned for the mission, we also compute the capture ΔV according to Equations 1 – 4.

$$r_p = r_{planet} + h_p \tag{1}$$

$$a = \sqrt[3]{\mu T^2/(4\pi^2)} \tag{2}$$

$$e = 1 - (r_p/a) \tag{3}$$

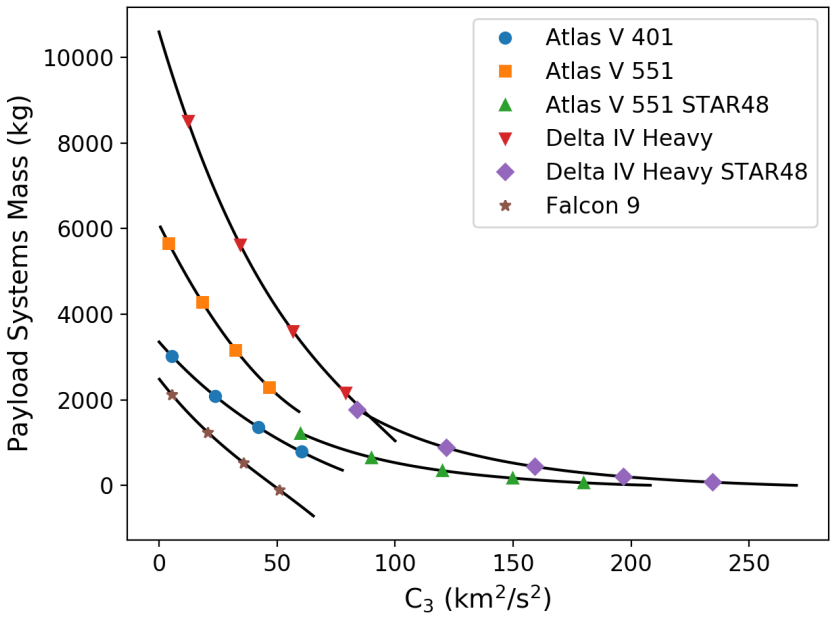


Figure 8. Some examples from the library of launch mass vs. C_3 curves maintained in the toolkit are plotted above. The toolkit determines the maximum launch mass from Earth for a particular launch V_∞ and launch vehicle using a performance curve like those above.

$$\Delta V_{capture} = \sqrt{V^2 + 2\mu/r_p} - \sqrt{\mu(1+e)/r_p} \tag{4}$$

where r_{planet} is the target planet radius drawn from a database within the toolkit, V is assumed to be the arrival V_∞ at the target planet provided by STOUR, and T and h_p are the period and periapse altitude of the capture orbit provided by the analyst. The capture ΔV is converted into a propellant mass as described above.

In this way, the initial mass is decremented to provide an estimate of the mass at each event in the mission. These steps are repeated for each trajectory solution and for each path considered. The procedure is illustrated in Figure 9. Here we see the mass histories for all the solutions that STOUR found along a single path. The dashed line highlights a single trajectory solution. Each of the solutions uses the same assumption for a launch vehicle (the same curve from Figure 8) and begins with a similar mass. The differences in initial mass result from the variation in launch V_∞. Notably, several solutions result in negative delivered mass under the given assumptions. These solutions can be eliminated from consideration. The toolkit performs the mass history calculations for all the examined paths which allows graphical comparison of all the potential trajectories based on the mass that the trajectory can deliver to the target planet.

Interactive Plots

In the Ice Giant catalog studies,[13,14] we (and others) have found the colorbar plot to be particularly useful. This type of plot represents two attributes of a trajectory on the X- and Y-axes and a third attribute on a graduated color bar as shown in Figure 10. Here, each point represents an individual trajectory from Earth to Uranus found by STOUR (with various intermediate flyby bodies).

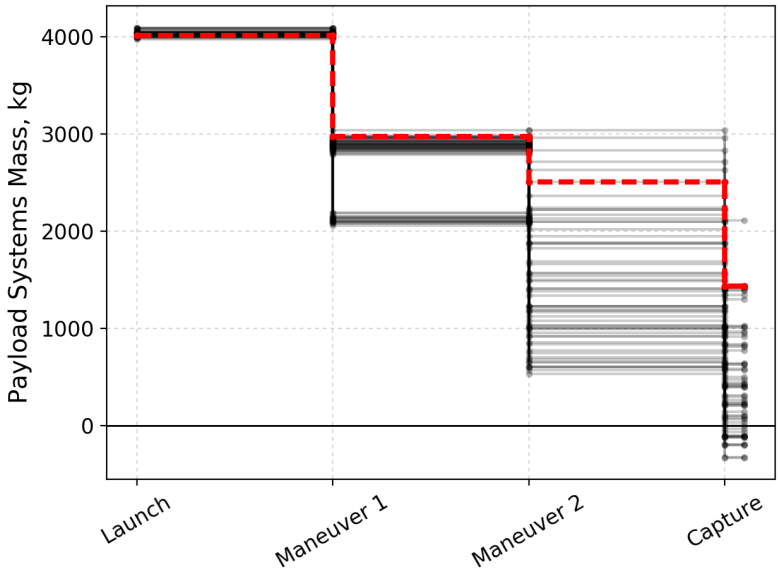


Figure 9. Mass histories derived from STOUR output are plotted above. The plot shows the spacecraft mass (on the y-axis) immediately after each of the mass change events (on the x-axis). The lines represent all the mass histories derived from a single STOUR solution file. The bold, dashed trace highlights a single trajectory among these solutions. The discontinuities in the lines represent the discrete change in mass at the mid-course and capture maneuvers. The process described here is repeated for each STOUR output file.

Trajectory designers quickly identify attractive trajectories using these plots. Appropriate selection of the plotting axes results in visible trends for the most desirable trajectories. For example, in Figure 10, the preferred family of trajectories lies along the lower-right boundary of the plotted points. For a given point on this boundary, trajectories to the left deliver less mass for the same flight time and trajectories above require longer flight time for the same delivered mass. In addition, lighter shaded points are less costly in terms of launch V_∞.

The colorbar may also represent categorical attributes of each trajectory. Figure 11 plots the time-of-flight against the launch date for the dataset shown in Figure 10. Here, we highlight the multiple paths to Uranus found by STOUR in the colorbar. For example, in Figure 11, the darkest shade represents the VVV0SU (Venus-Venus-Venus-Maneuver-Saturn-Uranus) path. By comparing the location of the VVV0SU points to the full dataset, we see that this path tends to yield longer flight times. We also note that plotting against launch date gives us insight into the periodic availability of the different paths. The color palette can be selected for grayscale printing or to exaggerate differences in the data.

The designer may click on a plot point to display detailed information for the selected trajectory. For example, while analyzing Figure 11, we may wish to find a low launch V_∞ or a specific launch-date range. This easy access to other (unplotted) attributes allows the designer to find the most attractive option from among several similar trajectories.

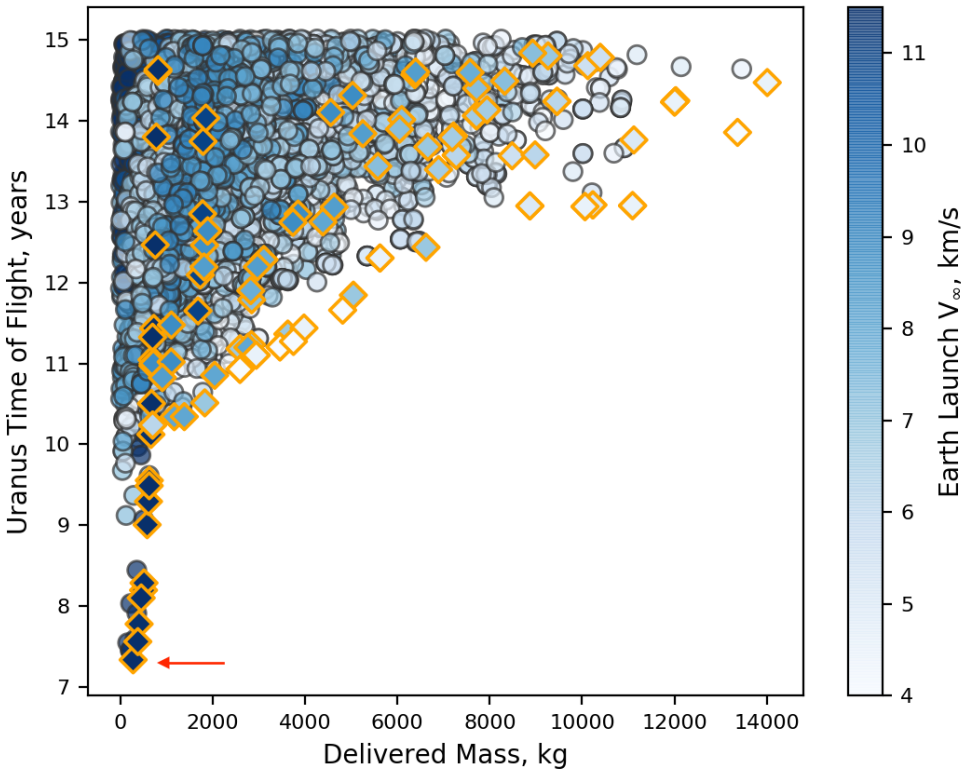


Figure 10. The output of our visualization tool is illustrated in this figure. Presented here are thousands of gravity-assist trajectories to Uranus. Among these trajectories is a case (indicated by the arrow) in which the time-of-flight is 7 years, the delivered mass is 300 kg, and the Earth launch V_∞ is 11 km/s. The diamonds highlight the Pareto-optimal set of minimum time-of-flight, minimum propellant mass, and maximum delivered mass. Figure 11 presents the same trajectories but shows the gravity-assist paths.

Automated Analysis

Establishment of a common database for the broad search results also enables our tool to automate analyses. Some examples are examined below. A Pareto front algorithm identifies non-dominated trajectories with respect to user-selected design objectives. Filtering methods remove trajectories that do not meet user-specified constraints (such as solutions outside of a particular span of launch dates). A cataloging process groups trajectories by launch date and identifies the "best" option (as defined by the user) in each group. Task-specific analyses can be scripted and performed multiple times.

Our toolkit includes a Pareto optimal sorting algorithm to assist analysts in finding trajectories that are desirable in multiple objectives. The sorting algorithm is adapted from the Simple Cull algorithm in Yukish[17] and is outlined in Algorithm 1 in the Appendix. The analyst provides a list of parameters to optimize. In our implementation, the parameters are variables to be minimized (or are cast as such). The parameters may be any field loaded or derived from STOUR in the DataFrame. The Pareto algorithm identifies those trajectories along the Pareto frontier for the given parameters. The Pareto frontier is optimal in the sense that no solution on the frontier is dominated in every objective by any other solution.

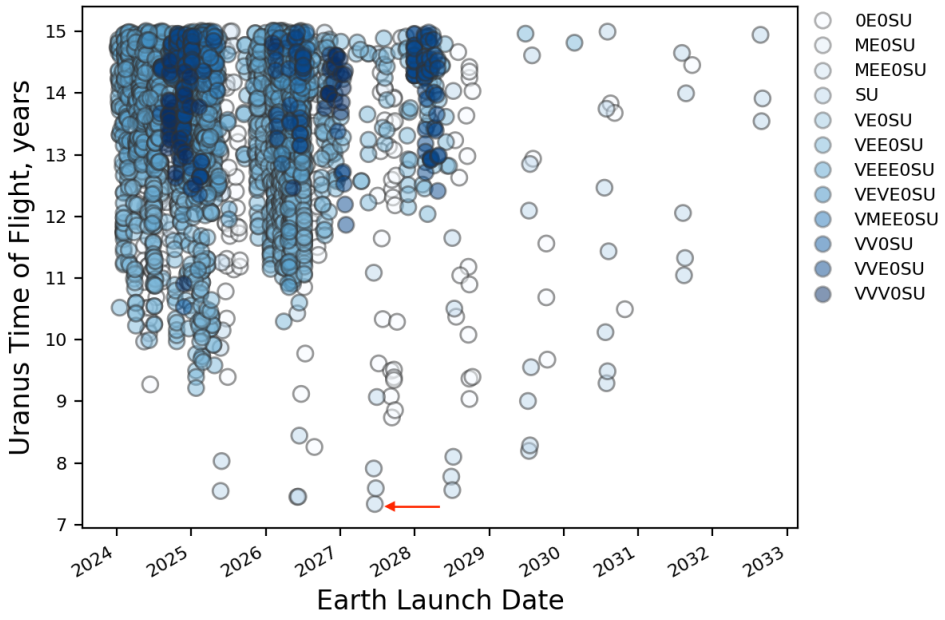


Figure 11. The Uranus trajectories in Figure 10 are replotted in this figure. However, in this example, the X-axis represents the launch date and the shade of the marker denotes the gravity-assist path for each trajectory. The case indicated by the arrow in Figure 10 is highlighted again here and follows the path SU (Saturn-Uranus).

In our context, $T_{i,1-m}$ in Algorithm 1 represents an individual conic solution. The algorithm first sorts the trajectories by the first objective and moves the first trajectory to the Pareto set. The remaining trajectories are then compared to the Pareto set. The procedure removes dominated trajectories which allows non-dominated trajectories to bubble to the top of the list. Once a trajectory reaches the top, it is added to the Pareto set (since it has not been dominated by any member already in the Pareto set). After the sorting has been performed, members of Pareto set can be highlighted in the toolkit plots or exported to an output file.

The diamonds in Figure 10 show an example of a Pareto-optimal set for minimum time-of-flight, maximum delivered mass, and minimum propellant mass. In this figure, the plotting axes are chosen for the first two objectives (time-of-flight and delivered mass). If the Pareto set were to be computed for only time-of-flight and delivered mass it would lie along the lower-right edge of the plotted points and plotting the Pareto set would be unnecessary. When three or more objectives are considered, the Pareto set can identify cases that would not be obvious in a two-dimensional plot. Figure 10 shows cases that are not along the lower-right boundary that are in the Pareto-optimal set because of the additional objective (minimizing propellant mass).

In addition to the Pareto filter, the toolkit provides some basic filtering rules that can quickly eliminate trajectories that are not interesting to the analyst. The simplest filter removes infeasible trajectories. Using the mass history derived from the launch vehicle and maneuvers, the toolkit identifies and removes trajectories that result in a negligible delivered mass at the target planet. A generic filtering function removes trajectories that do not satisfy user-specified limits on any parameter (or group of parameters) in the DataFrame. For example, a filter can be set to remove trajectories that exceed a certain propellant mass value, or that launch outside of a desired launch date range.

The toolkit's cataloging utility applies several user-configurable filters to reduce the complete dataset into a list of attractive trajectories in outlying years. The catalog utility first groups the trajectories into launch-time periods (e.g. groups of trajectories with the same year of launch). Within these groups, the utility limits results to those that exceed a user-defined, minimum delivered mass. Finally, from the reduced group of trajectories, the utility selects the trajectory with the minimum time-of-flight in each launch period.

STOUR applies a grid-search technique to find patched-conic solutions. A finer grid could produce better results for some design objectives. Trajectories that are optimized or propagated in a higher-fidelity simulation may produce different results. For example, trajectory optimizers should be able to improve delivered mass or reduce time-of-flight. The Pareto-optimal set produced directly from STOUR solutions may differ from the Pareto-optimal set produced from STOUR solutions that have been optimized or reproduced with higher fidelity. To account for this fact, the catalog utility has logic to include additional trajectories in a user-specified region around the primary trajectory. The resulting collection of trajectories can be tabulated for further study.

CONCLUSION

Our tool provides new and powerful analysis techniques for broad gravity-assist trajectory searches. Combining many potential paths into a single database enables side-by-side comparison of trajectory features across dissimilar trajectories. The plotting techniques allow analysts to spot trends in trajectory solutions that would not be evident in the traditional text output. The filtering and sorting methods deliver quick identification of attractive trajectories that warrant further investigation. The ability to add user-defined data to the STOUR database enables users to easily weigh competing trajectory solutions in aspects that may not be obvious from raw STOUR output. These capabilities have proven themselves useful to our research needs and may help to uncover new mission opportunities in the future.

REFERENCES

[1] E. A. Rinderle, "Galileo Users Guide, Mission Design System, Satellite Tour Analysis and Design Subsystem," Report JPL D-263, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, July 1986.

[2] J. Longuski and S. Williams, "Automated Design of Gravity-Assist Trajectories to Mars and the Outer Planets," *Celestial Mechanics and Dynamical Astronomy*, Vol. 52, No. 3, 1991, pp. 207–220.

[3] M. R. Patel, "Automated Design of Delta-V Gravity-Assist Trajectories For Solar System Exploration," Master's thesis, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, 1993.

[4] S. N. Williams and J. M. Longuski, "Low Energy Trajectories to Mars via Gravity Assist from Venus to Earth," *Journal of Spacecraft and Rockets*, Vol. 28, July 1991, pp. 486–488.

[5] J. Longuski and S. Williams, "The Last Grand Tour Opportunity to Pluto," *Journal of the Astronautical Sciences*, Vol. 39, 1991, pp. 359–365.

[6] M. R. Patel, J. M. Longuski, and J. A. Sims, "A Uranus-Neptune-Pluto Opportunity," *Acta Astronautica*, Vol. 36, No. 2, 1995, pp. 91–98.

[7] M. R. Patel, J. M. Longuski, and J. A. Sims, "Mars Free Return Trajectories," *Journal of Spacecraft and Rockets*, Vol. 35, No. 3, 1998, pp. 350–354.

[8] A. E. Petropoulos and J. M. Longuski, "Trajectories to Jupiter via Gravity Assists from Venus, Earth, and Mars," *Journal of Spacecraft and Rockets*, Vol. 37, No. 6, 2000, pp. 776–783.

[9] M. Okutsu and J. M. Longuski, "Mars Free Returns via Gravity Assist from Venus," *Journal of Spacecraft and Rockets*, Vol. 39, No. 1, 2002, pp. 31–36.

[10] C. M. Spreen, M. Mueterthies, K. Kloster, and J. Longuski, "Preliminary Analysis of Ballistic Trajectories to Uranus Using Gravity-Assists from Venus, Earth, Mars, Jupiter, and Saturn," *AAS/AIAA Astrodynamics Specialist Conference*, Girdwood, AK, July 31 - Aug. 4 2011.

[11] K. M. Hughes, J. W. Moore, and J. M. Longuski, "Preliminary Analysis of Ballistic Trajectories to Neptune via Gravity Assists from Venus, Earth, Mars, Jupiter, Saturn, and Uranus," *AAS/AIAA Astrodynamics Specialist Conference*, Hilton Head Island, SC, Aug. 11–15 2013.

[12] K. M. Hughes, P. J. Edelman, S. J. Saikia, J. M. Longuski, M. E. Loucks, J. P. Carrico, and D. A. Tito, "Fast Free Returns to Mars and Venus with Applications to Inspiration Mars," *Journal of Spacecraft and Rockets*, Vol. 52, No. 6, 2015, pp. 1712–1735.

[13] K. M. Hughes, *Gravity-Assist Trajectories to Venus, Mars, and the Ice Giants: Mission Design with Human and Robotic Applications*. Ph.d. dissertation, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, 2016.

[14] A. J. Mudek, J. W. Moore, K. M. Hughes, S. J. Saikia, and J. M. Longuski, "Ballistic and High-Thrust Trajectory Options to Uranus Considering 50 Years of Launch Dates," *AAS/AIAA Astrodynamics Specialist Conference*, Stevenson, WA, Aug. 20–24 2017.

[15] G. Rossum, "Python Reference Manual," tech. rep., Amsterdam, The Netherlands, 1995.

[16] W. McKinney, "Data Structures for Statistical Computing in Python," *Proceedings of the 9th Python in Science Conference*, Vol. 445, Austin, TX, June 28 - July 3 2010, pp. 51–56.

[17] M. A. Yukish, *Algorithms to Identify Pareto Points in Multi-dimensional Data Sets*. Ph.d. dissertation, College of Engineering, Pennsylvania State University, University Park, PA, 2004.

APPENDIX: PARETO ALGORITHM

Algorithm 1: Pareto Sorting Algorithm

function pareto_search (T);
Input : Table of data with n values for each of m objectives, $T_{1-n,1-m}$
Output: pareto set
P = empty array; // The Pareto set
sort T in ascending order of the first objective column;
// Until T is depleted
while *length(T)* > 0 **do**
  // Move the top row of T to the pareto set since it is not dominated
  append $T_{1,1-m}$ to P;
  remove $T_{1,1-m}$ from T;
  $i = 0$;
  // While T is not depleted and we have not stepped all the way through
  // Check each row of T against the latest pareto set addition
  while *length(T)* $\neq 0$ *and* $i <$ *length(T)* **do**
    // For each objective
    dominated = 0;
    // Step through the columns
    for $c = 1$ *to* m **do**
      // Count columns that are dominated by the latest pareto set addition
      if $T_{i,c} > P_{end,c}$ **then**
        dominated = dominated + 1;
      end
    end
    if *dominated* = *m* **then**
      // Discard this row because it is dominated in each column
      remove $T_{i,1-m}$ from T;
    else
      // Continue to the next row
      $i = i + 1$
    end
  end
end
return P;