Unified Lambert Tool for Massively Parallel Applications in Space Situational Awareness

Robyn M. Woollands[1], Julie Read[2], Kevin Hernandez[3], Austin Probe[4], John L. Junkins[5]
Department of Aerospace Engineering, Texas A&M University.

This paper introduces a parallel-compiled tool that combines several of our recently developed methods for solving the perturbed Lambert problem using modified Chebyshev-Picard iteration. This tool (unified Lambert tool) consists of four individual algorithms, each of which is unique and better suited for solving a particular type of orbit transfer. The first is a Keplerian Lambert solver, which is used to provide a good initial guess (warm start) for solving the perturbed problem. It is also used to determine the appropriate algorithm to call for solving the perturbed problem. The arc length or true anomaly angle spanned by the transfer trajectory is the parameter that governs the automated selection of the appropriate perturbed algorithm, and is based on the respective algorithm convergence characteristics. The second algorithm solves the perturbed Lambert problem using the modified Chebyshev-Picard iteration two-point boundary value solver. This algorithm does not require a Newton-like shooting method and is the most efficient of the perturbed solvers presented herein, however the domain of convergence is limited to about a third of an orbit and is dependent on eccentricity. The third algorithm extends the domain of convergence of the modified Chebyshev-Picard iteration two-point boundary value solver to about 90% of

---

[1] Postdoctoral Research Associate, Department of Aerospace Engineering, Texas A&M University, TAMU 3141, College Station, TX 77843-3141, USA.
[2] PhD Graduate, Department of Aerospace Engineering, Texas A&M University, TAMU 3141, College Station, TX 77843-3141, USA.
[3] PhD Candidate, Department of Aerospace Engineering, Texas A&M University, TAMU 3141, College Station, TX 77843-3141, USA.
[4] PhD Candidate, Department of Aerospace Engineering, Texas A&M University, TAMU 3141, College Station, TX 77843-3141, USA.
[5] Distinguished Professor, Department of Aerospace Engineering, Texas A&M University, TAMU 3141, College Station, TX 77843-3141, USA.

an orbit, through regularization with the Kustaanheimo-Stiefel transformation. This is the second most efficient of the perturbed set of algorithms. The fourth algorithm uses the method of particular solutions and the modified Chebyshev-Picard iteration initial value solver for solving multiple revolution perturbed transfers. This method does require "shooting" but differs from Newton-like shooting methods in that it does not require propagation of a state transition matrix. The unified Lambert tool makes use of the General Mission Analysis Tool and we use it to compute thousands of perturbed Lambert trajectories in parallel on the Space Situational Awareness computer cluster at the LASR Lab, Texas A&M University. We demonstrate the power of our tool by solving a highly parallel example problem, that is the generation of extremal field maps for optimal spacecraft rendezvous (and eventual orbit debris removal). In addition we demonstrate the need for including perturbative effects in simulations for satellite tracking or data association. The unified Lambert tool is ideal for but not limited to space situational awareness applications.

I. Introduction

Lambert's problem is the classical two-point boundary value problem (TPBVP) in celestial mechanics, that was first posed and solved by Johann Heinrich Lambert in 1761. It is known to have a unique solution for the fractional orbit transfer between prescribed positions in a prescribed "time of flight". Solving the problem requires determining the orbital arc (typically, solving for the initial velocity) connecting a prescribed initial position and final position, which correspond to the specified flight time. In the modern literature, Richard Battin [1, 2] developed an immortal and the most widely used and general algorithm for solving the unperturbed Lambert's Problem (Keplerian motion). His algorithm generates not only the unique solution for the fractional orbit case, but also the multiple solutions associated with multiple revolution orbit transfers. Prussing further refined this method and published an elegant formulation and algorithm in 1992 [3].

The most common solution approach for generalizing the Lambert problem to include pertur-

bations is to utilize the state transition matrix sensitivity of the final state with respect to the initial velocity, and iterate via Newton's method on the three components of initial velocity to "hit" the final desired position at the prescribed final time. Another method for solving the generalized Lambert's problem that is not as well-known but is also very good is the method of particular solutions (MPS)[4]. MPS makes assumptions based on local-linearity and utilizes a reference trajectory and a set of particular solutions. Variations of the initial velocity to the position of the particular solutions at the final time are used to generate a subspace containing the current "miss vector" at the final time. A linearity assumption for the departure of the desired particular solution relative to the reference trajectory permits an update of the initial velocity via a Newton-type iteration. In both cases the unperturbed Lambert solution can be used as a "warm start" to solve the perturbed problem.

The focus of this paper is the development of the unified Lambert tool (ULT) that consists of three algorithms that solve the perturbed Lambert problem and one that solves the Keplerian problem. The algorithms accommodate state of the art force models and the tool as a whole is implemented in C/C++ and in parallel, using Message Passing Interface (MPI), for high performance computation on a 192 core computer cluster. Each algorithm that forms part of the ULT is discussed in the section following, with relevant references provided where more details can be found.

One motivation for this research is to respond to the various challenges in Space Situational Awareness (SSA) with a difficult "data association" problem. Short tracks of many newly observed objects, widely separated in time, must be processed to determine orbits and correlate the observations of tracked objects, if possible, with each other and with existing space object data bases. In the current state of the practice, hundreds of thousands of hypotheses must frequently be tested to find feasible preliminary orbits connecting time-displaced short tracks of unknown space objects. These preliminary orbits and the underlying data associations are taken as the starting estimates for further correlation. "Short" tracks may be separated by up to several orbits, so ignoring the effects of perturbations will typically introduce residual errors much larger than the measurement errors, which can corrupt the data association process.

Data association hypotheses are currently tested for preliminary orbit estimation using the Keplerian Lambert solutions for sufficiently short arcs, but higher precision is needed to accommodate hypothesis testing over longer time intervals. When more than several hundred thousand hypotheses are tested daily (including perturbations), the computational cost can exceed many CPU days per month. The anticipation of a new space fence giving an order of magnitude increase to $\sim 200,000$ more presently un-trackable debris objects (visible to our sensors) means that already high computational costs are about to dramatically increase [5]. Millions of hypotheses will require testing to solve the data association problems. Also, "all-on-all" conjunction analysis and probability of collisions will be extremely difficult using existing orbit propagation tools. Thus the issue of finding an accurate and efficient solution to a generally perturbed TPBVP lies near the heart of many computational challenges in SSA. The inclusion of perturbations in Lambert's problem and the development of more efficient and robust methods are therefore of strong interest.

In addition to the data association problem, which deals with tracking, there is also the problem of debris removal that must be considered. The satellite collision of Iridium and Kosmos, in 2009, demonstrated the seriousness of the orbit debris problem. In an instant hundreds of thousands of fragments with much higher than bullet speeds began orbiting the Earth. This debris is hazardous to operational satellites and reducing the risk of future collisions is possible by rendezvous, capture and de-orbit missions to remove the largest derelict objects. There are over 500 USA-launched spent rocket boosters in low Earth orbit [6]. Determining the globally optimal sequence of maneuvers for retrieving orbital debris can require simulating thousands of transfer trajectories. The Δv cost for each must be computed and displayed in an extremal field map (EFM) in order to effectively distinguish globally optimal from infeasible and sub-optimal orbit maneuver regions. We present a case study that illustrates the power of the ULT by generating EFMs in parallel, and we also demonstrate the importance of including perturbations in data association analysis.

II. Unified Lambert Tool

The ULT is a parallel C/C++ algorithm that may be used to solve for a set of transfer trajectories between two specified sets points and the associated times-of-flight (data association type

problem), or to simulate transfer trajectories between two orbits (EFM type problem). In each case, the simulation of hundreds of thousands of trajectories may be required.

A. Sub-Algorithms

The ULT consists of four Lambert algorithms written in a C/C++ environment and a suite of MATLAB post-processing tools for plotting and analysis. A schematic of the ULT is shown in Figure 1. Each Lambert problem that the ULT is tasked with solving is first computed with the Keplerian solver [3], and then if a perturbed trajectory is required the ULT automatically selects the best suited perturbed algorithm for the job. There are three perturbed algorithm choices, and the arc length or true anomaly angle spanned by the Keplerian transfer trajectory is the parameter that governs the automated selection of the appropriate perturbed algorithm. This selection is based on the algorithm convergence characteristics and efficiency of the respective perturbed solvers. More details on the modified Chebyshev-Picard iteration (MCPI) algorithm convergence may be found in the Bai's [7] and Woollands' [8] PhD dissertations.

The Keplerian Lambert solver utilized in the ULT is limited to the computation of elliptic orbits only (no parabolic or hyperbolic orbits). This work was originally developed by Battin and published in his book, Astronautical Guidance, in 1964 [2]. Later Prussing, a student of Battin, revisited this work and published his additions and refinements to the method in 1992 [3]. Both algorithms (Battin's and Prussing's) consider the multiple possible solutions associated with multiple revolution Lambert problem. Prussing's version of the algorithm was adopted and re-coded in C and MATLAB for use in the ULT.

The first of the three perturbed Lambert solvers is the MCPI-TPBVP. This algorithm solves the TPBVP without using a shooting method whereby the linearly contained Chebyshev coefficients of the acceleration and the ensuing trajectory (that is iteratively converging to the solution) are formulated in such a way to implicitly determine the unknown initial velocity (constant of integration) using knowledge of the initial and final position only. This is the most efficient of the three perturbed solvers but has the drawback that convergence is limited to about a third of an orbit and the convergence degrades with increasing eccentricity. A brief explanation of the MCPI algorithm

is given in Appendix A, with a comprehensive expalnation and mathematical development for the MCPI-TPBVP solver presented in Appendix B.

The second of the perturbed algorithms is MCPI-TPBVP regularized with the Kustaanheimo-Stiefel (KS) transformation. This method, MCPI-KS-IVP, makes use of the KS time transformation [9] as well as a state variable change that rigorously linearizes the two-body problem through a judicious coordinate transformation. Woollands, et al. developed the MCPI-KS-TPBVP algorithm in 2015 [10]. The full derivation of the algorithm is presented in Appendix E of Woollands' PhD dissertation [8]. Regularization of the equations of motion when solved using the MCPI-TPBVP alters the convergence properties of the problem and as discussed in [10] increases the domain of convergence of the MCPI-TPBVP from about a third of an orbit to 90% of an orbit. Theoretically the MCPI-KS-TPBVP should converge for a full orbit, however as this upper bound is approached the time for the algorithm to converge approaches infinity, and as a practical matter we obtain convergence up to about 90% of an orbit (independent of eccentricity). The MCPI-KS-TPBVP is the second most efficient of the perturbed Lambert solvers in the ULT. This algorithm promises many applications, for example mid-course guidance, wherein no state transition matrix is required.

The third of the perturbed solvers uses MPS [4] and the MCPI initial value problem (IVP) solver. This algorithm, MPS-MCPI-IVP, is a shooting-type method that is a local-linearity based approach that requires a reference trajectory $\boldsymbol{r}_{ref}(t), \dot{\boldsymbol{r}}_{ref}(t), \ddot{\boldsymbol{r}}_{ref}(t)$ and a set of particular solutions. Variations of the initial velocity to generate particular solutions neighboring the nominal trajectory, for sufficiently small displacements from the reference trajectory, span all positions of the solutions at the final time. This permits the initial velocity to be updated via a Newton-type iteration. The mathematical development for this method is summarized in [11]. The MPS-MCPI-IVP algorithm converges over multiple revolutions, unlike the previous two methods, but it is the most expensive of the three algorithms in the ULT. [11] showed that MPS is at least competitive if not better than the regular Newton shooting method with regard to efficiency, while maintaining the comparable accuracy.

B. Parallelization

The ULT is implemented in parallel, using MPI, on the 192 core SSA computer cluster at the LASR Lab, Texas A&M University. The nature of the parallelization is general in the sense that the ULT can compute multiple trajectories at any instant in time if they are independent, but it is specific in the sense that a unique "front end" to the ULT is required for solving different types of problems.

For example, to solve a large set of Lambert problems (data association type problem) the ULT will accept a configuration file with a user specified number of position boundary conditions and the corresponding user speficied times-of-flight. All the computations of these potential Lambert transfers will be computed in parallel until the job is completed. If for example, an EFM is required then the parallelization is slightly different as there are two possible layers of parallelization that may be utilized. The first layer involves the number of revolutions (see [3], where 2π is added for each additional revolution) and the second layer of parallelization is related to each "vertical row" (increasing time-of-flight) on the EFM. In both cases the only limitation is the number of compute nodes available for performing the task.

C. User Input

The user input required by the ULT differs slightly for solving different types of problems. Below is a brief description of the initial conditions and simulation parameters required in the configuration file(s). More details are given in the user manual and code.

1. Keplerian OR Perturbed final solution?
2. Single Trajectory OR Set of Trajectories OR EFM?
    (a) Single trajectory
        i. Specify initial and final position (Cartesian or classical elements).
        ii. Specify the desired time-of-flight.
    (b) Set of Trajectories
        i. Specify *all* initial and final positions (Cartesian or classical elements).

ii. Specify *all* the corresponding times-of-flight.

(c) EFM

i. Specify departure and arrival orbit (Cartesian or classical elements).

ii. Specify EFM dimensions.

A. Maximum time-of-flight (y-axis)

B. Maximum time past arbitrary starting point (x-axis)

3. Force Model

(a) Specify spherical harmonic degree and order

(b) Specify atmospheric drag parameters

(c) Specify third body effects

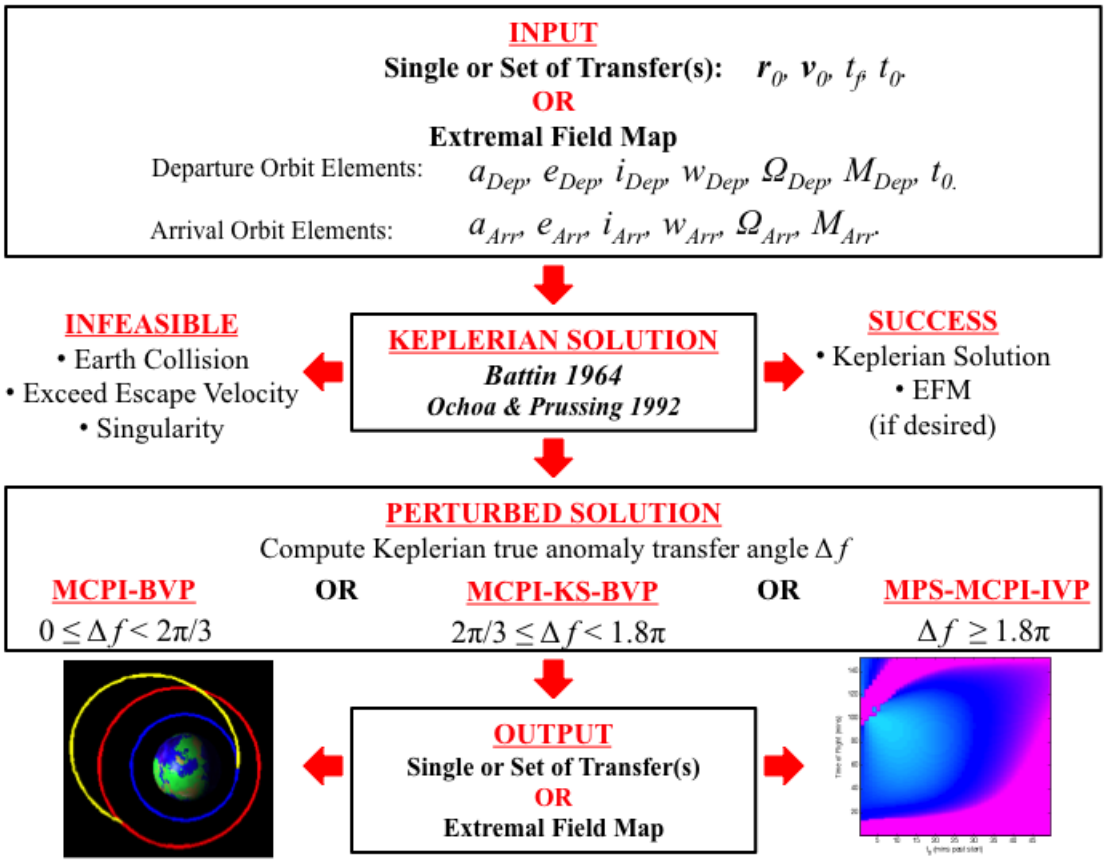


Fig. 1 Flow diagram outlining the components in the unified Lambert tool.

III. Parallel Generation of Extremal Field Maps

Orbit debris is not distributed uniformly in space, in fact forty percent of the debris in LEO is located in nine distinct clusters. These clusters contain anywhere from 20 to 150 spent rocket boosters and the bulk of the booster mass is located at inclinations of about 65°, 71°, 74°, 81° and 83° [12]. Two smaller clusters are near 28° and 98°. In addition to these clusters of spent boosters, there are thousands of other small debris objects spanning a range or inclinations. Spent boosters are attractive objects to remove because they all represent potential collisions and removing these large objects has been found to be one of the best ways to mitigate debris growth. Since removing several of these boosters on a given launch is a nonlinear "orbital traveling salesman" optimization problem, hundreds of thousands of iterative trajectories may need to be accurately and efficiently generated to determine the optimal trajectory sequence for rendezvous, docking and de-orbit of these boosters.

The ULT is used to simulate transfer trajectories between a pair of LEO orbits that could represent a retrieval spacecraft seeking to find the optimal maneuver for rendezvous with a piece of orbit debris. Since the debris has typically been in orbit for decades, this is not a fixed time problem. The take-off time is unknown and the time of flight between successive captures may entail many orbits. For each transfer the Δv cost is computed and the results are displayed on an EFM. Each EFM is composed of thousands of possible transfer trajectories, and presenting the data in this way allows mission design engineers to usually distinguish globally optimal from infeasible and sub-optimal orbit maneuver regions.

A. Multiple Solutions: Possible vs Feasible

There are $2N + 1$ solutions to the multiple revolution Lambert's problem [2, 3]. Figures 2 and 3 are two example test cases that show the multiple ways a spacecraft can transfer from a certain position on the blue orbit to a certain position on the red orbit in a given time of flight. In each of these figures the legend communicates the number of revolutions (N) and also whether the transfer resides on the *upper* or *lower* branch. This terminology is defined and discussed in both [3] and [11]. It is not essential for the reader to have a deep mathematical understanding of the

upper and *lower* branch classifications for understanding the material that follows in this paper, but it is necessary to recognize that multiple solutions do exist and these must be considered when generating EFMs. Note that these figures reveal a number of mathematically possible solutions but not all are physically feasible. Both these examples are for planar orbit transfers and it is clear that some trajectories collide with the Earth.

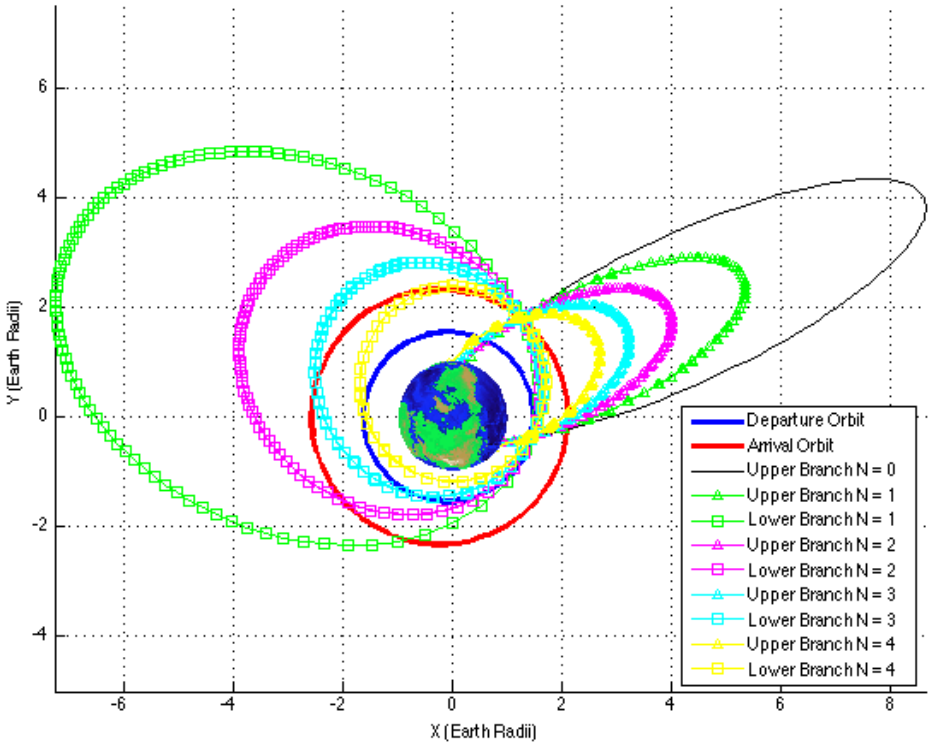


Fig. 2 Possible transfer trajectories between the departure (blue) and arrival (red) orbits. All transfer trajectories have the same time-of-flight and different semimajor axis values, hence each is undergoing some number of revolutions about the Earth. This set of transfers corresponds to the black dot in the EFMs that follow.

Over the next few pages a series of figures are presented that display all the possible solutions (left panels) for an orbit transfer between the two specified low Earth orbits, and of those, the ones that are actually physically feasible (right panels). The axes on these figures are departure time past some arbitrary starting time on the x-axis and time-of-flight on the y axis. The integer number of solutions is represented using various colors. Figure 4(a) and (b) show the number of solutions

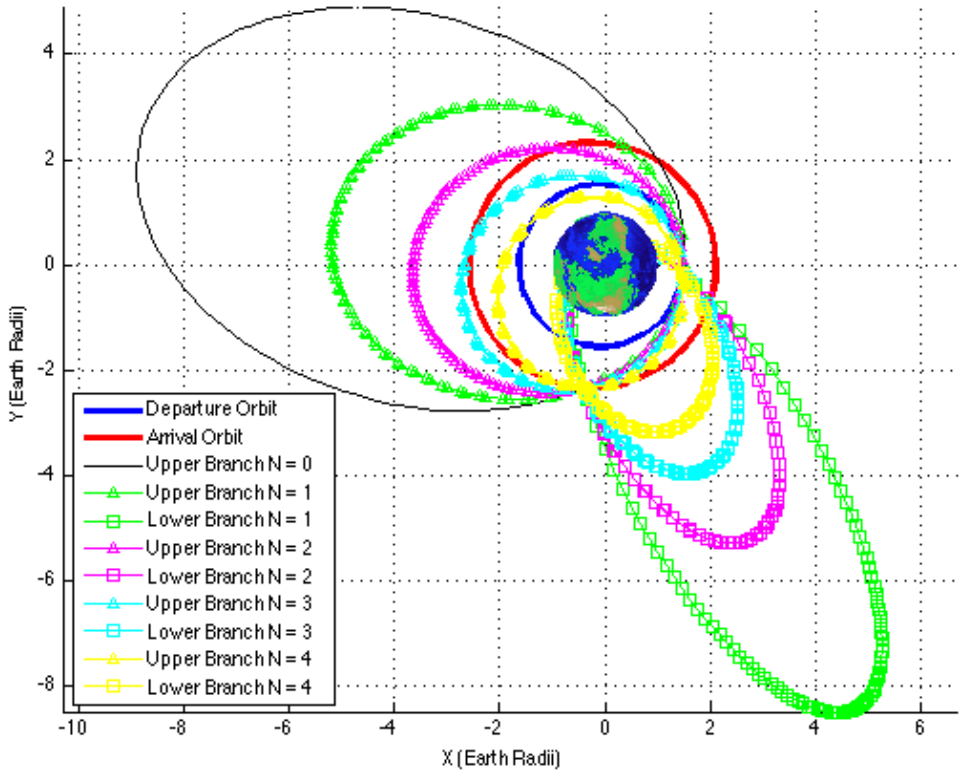
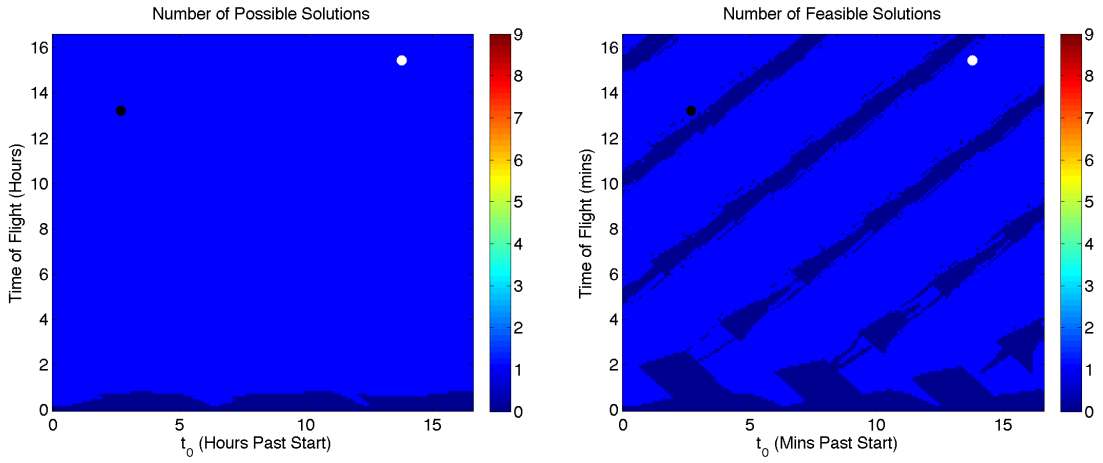


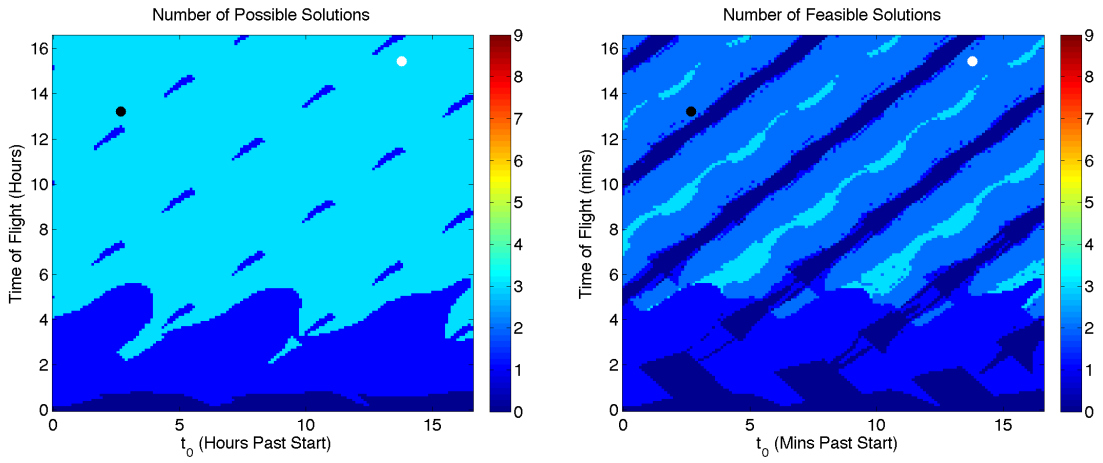
Fig. 3 Possible transfer trajectories between the departure (blue) and arrival (red) orbits. All transfer trajectories have the same time-of-flight and different semimajor axis values, hence each is undergoing some number of revolutions about the Earth. This set of transfers corresponds to the white dot in the EFMs that follow.

for transfers spanning a true anomaly angle range of $0 < f < 2\pi$. This is noted as the $N = 0$ revolution. As is evident, most of the EFM (a) shows that a transfer is "possible" except for a few small regions at the bottom where the time-of-flight is too short to make an elliptic transfer. In contrast, EFM (b) shows the feasible transfers. Note that there are more dark blue regions where a transfer is infeasible. This is due to the fact that certain transfers in the simulation collide with the Earth making them infeasible candidates. Note also the black and white dots in these figures, previously mentioned, that represent the set of transfers shown in Figures 2 and 3 respectively.



(a) Possible solutions (N=0) **(b) Feasible solutions (N=0)**

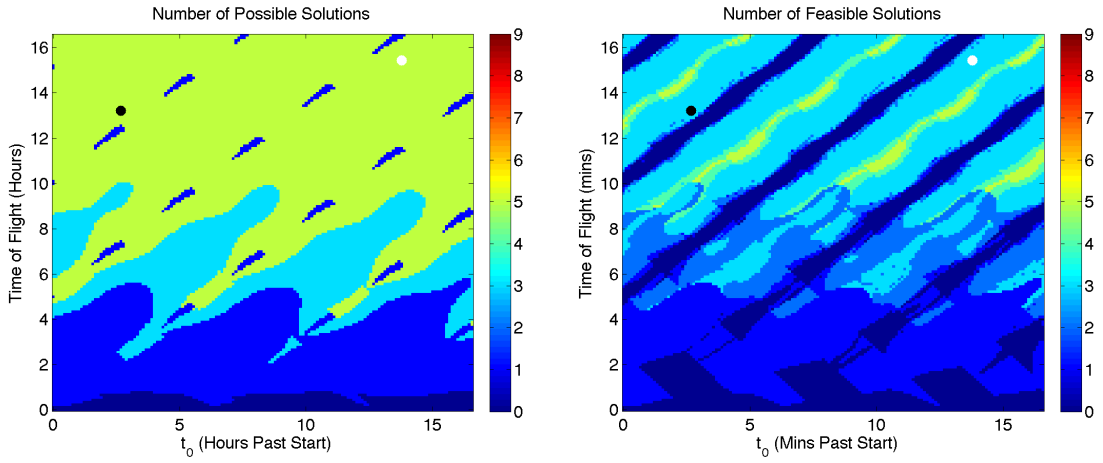
Fig. 4 Number of possible solutions versus feasible solutions for N=0.



(a) Possible solutions (N=1) **(b) Feasible solutions (N=1)**

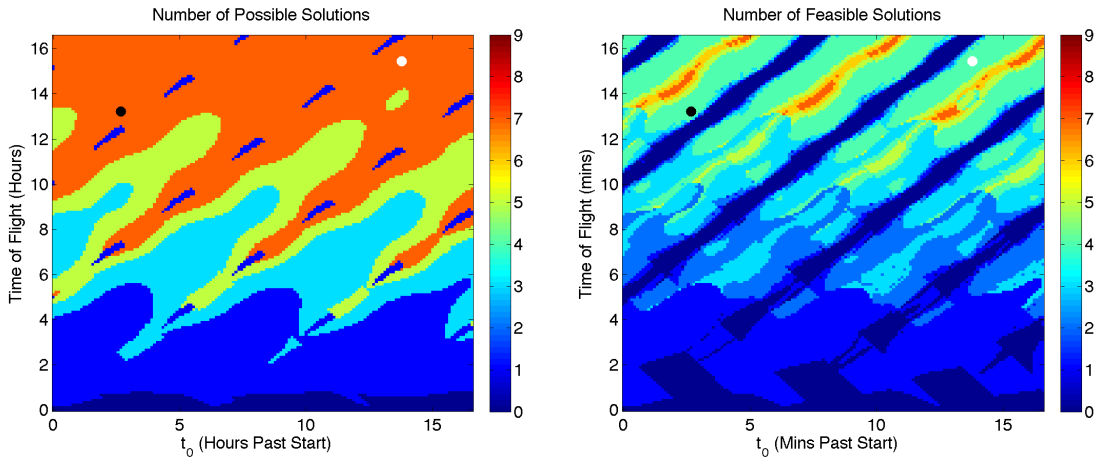
Fig. 5 Number of possible solutions versus feasible solutions for N=1.

Figure 5(a) and (b) show the the number of solutions for transfers spanning a true anomaly angle range of $0 < f < 4\pi$. Note that the maximum number of solutions is $2N + 1 = 3$. The EFM (a) shows regions where 1, 2 or 3 solutions are possible, and the EFM (b) shows regions where 1, 2 or 3 solutions are feasible. The two new solutions corresponding to the $N = 1$ revolution have different semimajor axis values and in certain cases one may collide with the Earth and become an infeasible candidate, as is evident in Figures 2 and 3. These sets of solutions correspond to the two "branches" of solutions as discussed by [3, 11].



(a) Possible solutions (N=2) **(b) Feasible solutions (N=2)**

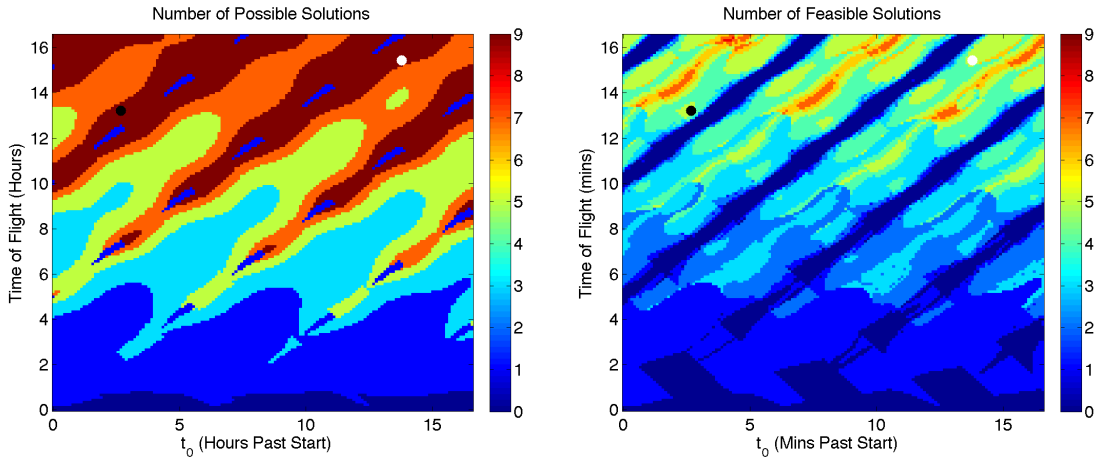
Fig. 6 Number of possible solutions versus feasible solutions for N=2.



(a) Possible solutions (N=3) **(b) Feasible solutions (N=3)**

Fig. 7 Number of possible solutions versus feasible solutions for N=3.

Figures 6 through 8 show the number of possible (a) and feasible (b) solutions for transfers spanning a true anomaly angle range of $0 < f < 6\pi$, $0 < f < 8\pi$ and $0 < f < 10\pi$ respectively. The maximum number of possible solutions in EFM Figure 8(a) is $2N + 1 = 5$. The total number of possible solutions for this particular simulation is actually 19, although we have not included figures all the way up to 19. In addition, most of those additional higher N "possible" candidates are physically infeasible solutions and thus we find in this case that the total number of feasible solutions is the same as shown in Figure 8(b).



(a) Possible solutions (N=4) **(b) Feasible solutions (N=4)**

Fig. 8 Number of possible solutions versus feasible solutions for N=4.

B. Trajectory Cost Quantification

The Δv cost associated with each feasible transfer solution is computed and the results are presented in a Δv EFM. Magenta represents all infeasible transfers, either that an elliptic transfer could not be computed, a simulated transfer collided with the Earth, or that the Δv exceeded the maximum allowable value that we adopted in this study. In this simulation we arbitrarily selected a maximum allowable Δv of 10 km/s. The various shades of blue represent feasible transfers that become increasingly more expensive as the shade of blue darkens (see colorbar). Note the black dot, corresponding to the black transfers in Figure 2, lies in an infeasible region as it exceeded the maximum allowable Δv. The white dot, corresponding to the black trajectory in Figure 3, is a feasible transfer.

Figure 10 shows the Δv cost and infeasible transfer regions for $N = 1$. The *upper* and *lower* branch EFMs are shown in (a) and (b) respectively. Observe that for the *upper* branch EFM (a) the black dot, corresponding to the "green triangle" trajectory in Figure 2, lies in an infeasible region. This is because the trajectory collides with the Earth. On the *lower* branch, EFM (b), the transfer is feasible. That is the "green square" trajectory in Figure 2 is a feasible transfer. The reverse situation occurs with the white dots, and the result can be easily verified by referring to Figures 2 and 3.

Figures 11 through 13 show the Δv cost and infeasible maneuver regions for larger values of N. Finally in Figure 14(a), the minimum Δv values from each of the $2N+1$ EFMs are extracted and used to construct the "combined" Δv EFM. This is the plot that is most valuable to the user and it allows them to investigate potential maneuver regions that correspond to low Δv cost. In addition, (b) is a "combined" EFM for a set of orbits with the same orbital elements as in (a) but with an inclination change of 40°. As expected, the overall Δv required for transferring between these orbits is greater due to the required plane change. The final plot in this section (Figure 15) reveals which of the three perturbed solvers were called by the ULT for computing the transfers. Red represents the MCPI-TPBVP, yellow represents the MCPI-KS-TPBVP and blue represents the MPS-MCPI-IVP. The MATLAB post-processing tools allow the user to select which information they require to be plotted, if any, or simply to output the information for the optimal (minimum Δv) transfer. More details are given in the code and in the ULT user manual.

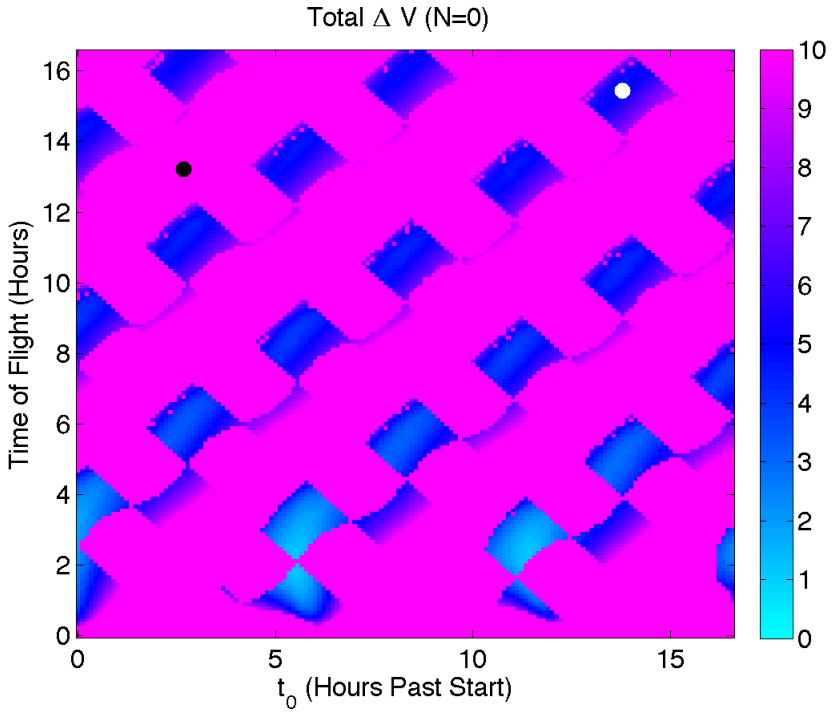
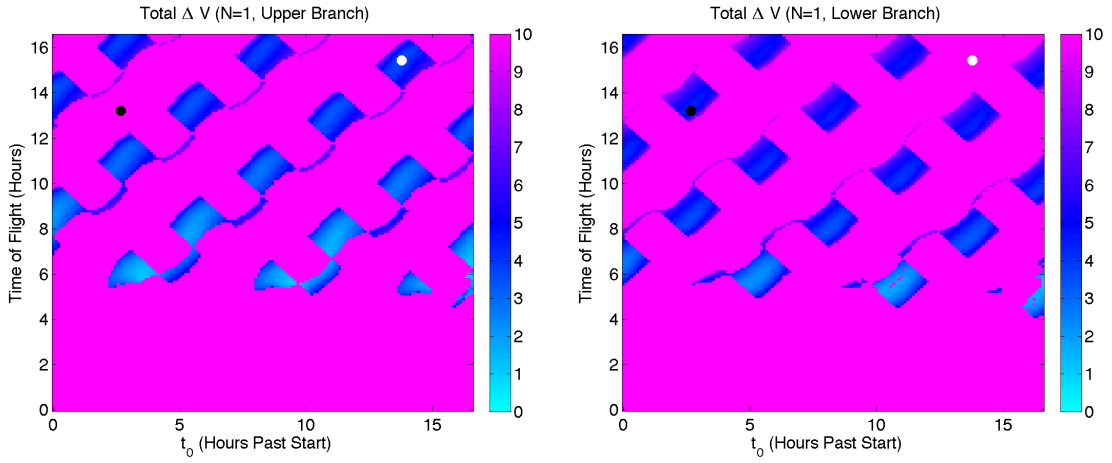
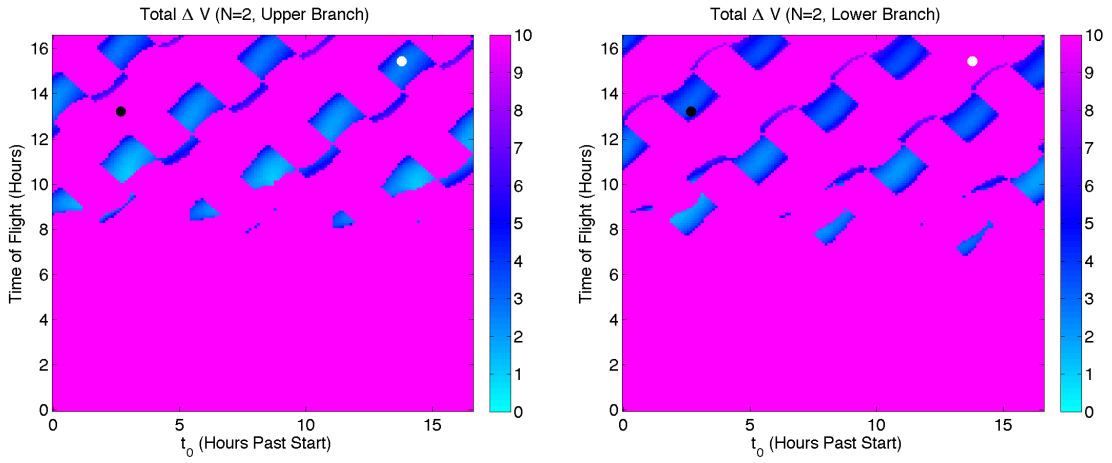


Fig. 9 Δv extremal field map for N=0 transfers.



(a) Upper Branch (N=1) **(b) Lower Branch (N=1)**

Fig. 10 Δv extremal field maps for N=1 transfers.

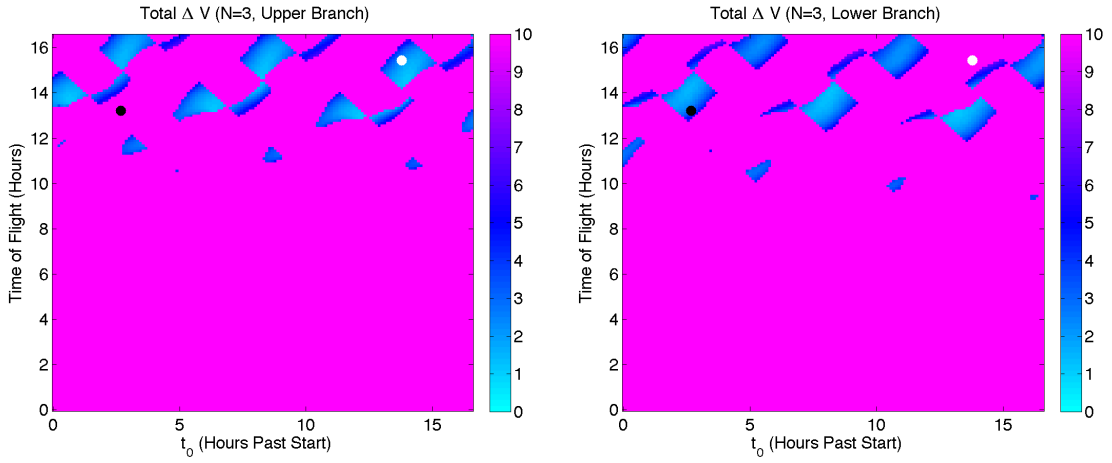


(a) Upper Branch (N=2) **(b) Lower Branch (N=2)**

Fig. 11 Δv extremal field maps for N=2 transfers.

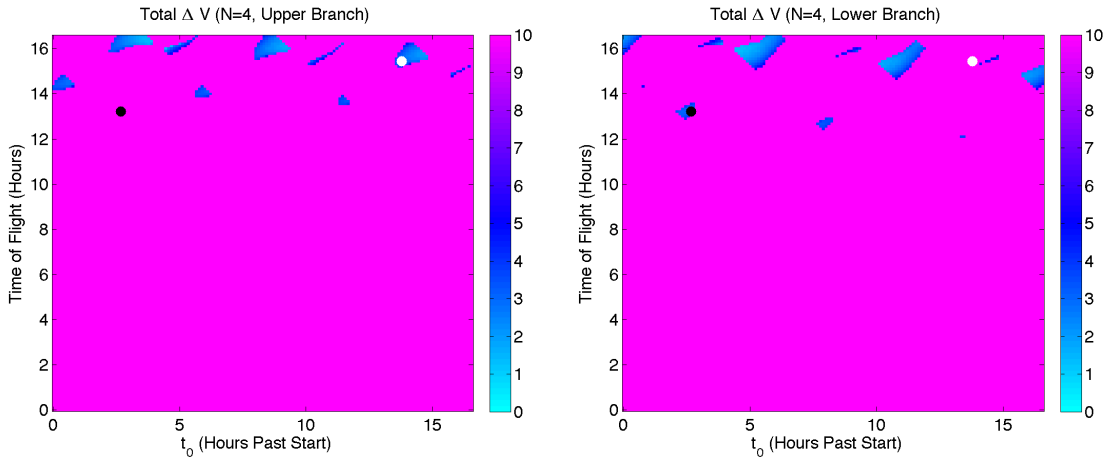
IV. Data Association Application

The U.S. Air Force radar fence that is currently under development will increase the number of trackable objects in low Earth orbit by an order of magnitude (about 20,000 objects to 200,000 objects) [5]. This increased volume of data and the increased statistical ambiguity associated with establishing orbits from the tracking measurements of this population will pose a computational grand challenge for building and maintaining a space object catalog. In this section the ULT is used to address the challenge by demonstrating the importance of including perturbations in data association algorithms. For this study the simulations are performed considering the deterministic Lambert problem only. In reality, estimation theory and uncertainty propagation are key elements in



(a) Upper Branch (N=3) **(b) Lower Branch (N=3)**

Fig. 12 Δv extremal field maps for N=3 transfers.

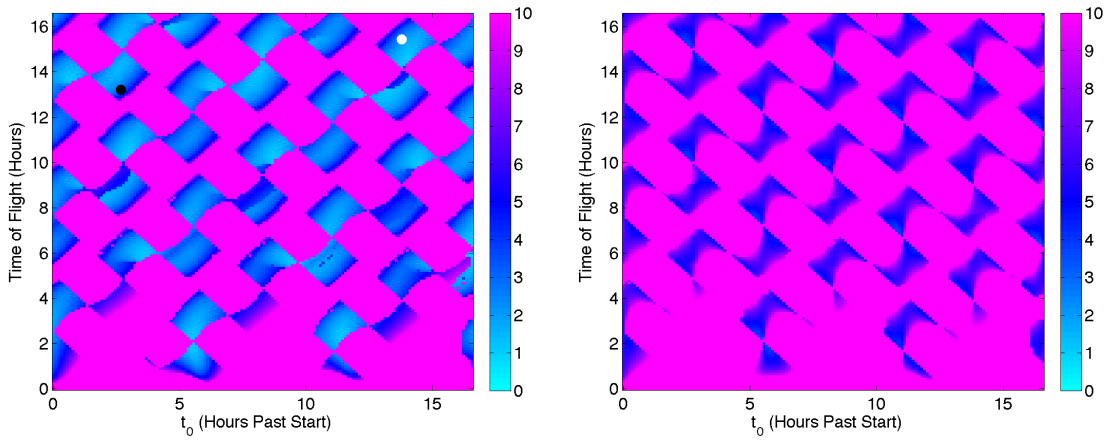


(a) Upper Branch (N=4) **(b) Lower Branch (N=4)**

Fig. 13 Δv extremal field maps for N=4 transfers.

state-of-the-art data association algorithms, as these are required for determining accurate physical approximations of the object prior to solving Lambert's problem. These aspects of the problem are beyond the scope of this paper and the work presented demonstrates the effectiveness of ULT, and its potential use as an asset to existing operational state-of-the-art data association algorithms.

Radar measurements usually provide the range, elevation, azimuth, and sometimes range-rate of the object relative to the observer location. If all these measurements are available then geometry and trigonometry can be used to extract a position and velocity from this data [13]. Sensor limitations sometimes provide only range, or only range and range-rate measurements. If this occurs then different techniques must be employed for determining the initial position and velocity guess [13].



(a) Inclination change of $0°$. **(b) Inclination change of $40°$.**

Fig. 14 "Minimum" Δv extremal field maps for combined (all N) transfers.

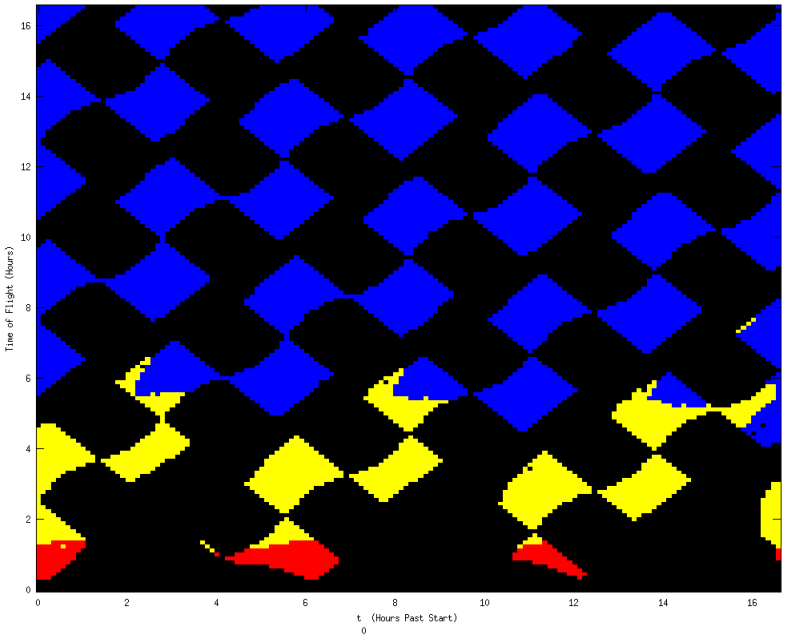


Fig. 15 Algorithm identifier: red represents MCPI-TPBVP, yellow represents MCPI-KS-TPBVP, blue represents MPS-MCPI-IVP

Orbit determination is not the focus of this paper and thus the assumption is made that adequate information is available from the sensors and that position and velocity is already computed and available for use in the ULT.

We performed a set of simulations to compare the solution of Lambert's problem with and with-

out gravity perturbations. Figures 16 and 17 show the maximum absolute position and velocity error along each particular Lambert solution trajectory. A 70^{th} degree and order spherical harmonic gravity field (EGM2008) is used for propagating the perturbed trajectories. All trajectories preserved the Hamiltonian to 14 digits and meet the final boundary conditions to sub-milimeter precision. Five test cases were done with semimajor axis values of 8000 km, 10,000 km, 20,000 km, 30,000 km and 42,000 km (typical geostationary orbit), eccentricity of zero and inclination of 10°. As expected, Figures 16 and 17 reveal that with increasing time (and arc length) over which the Lambert problem is solved there is an increase in the error between the perturbed and Keplerian Lambert solutions. The errors are also larger for orbits with smaller semimajor axis values where the trajectory is experiencing greater variations in the gravity field.

It is clear from these results that neglecting gravity perturbations when propagating trajectories and solving Lambert's problem will introduce considerable errors (as much as 10 km) that will lead to large uncertainties during the data association process. Including other perturbations such as drag, solar radiation pressure and third body effects would further reveal the discrepancies between the perturbed and Keplerian solutions. The parallel structure of the ULT and ability to efficiently include state of the art force models makes this tool significant and promising for near-term transition to establish a new state of the practice.

V. Conclusion

We have developed a parallel-compiled tool that combines several of our recently developed methods for solving the perturbed Lambert problem using modified Chebyshev-Picard iteration. This tool is named the *unified Lambert tool* and consists of four individual algorithms, each of which is unique and better suited for solving a particular type of orbit transfer. The first is a Keplerian Lambert solver that provides a good initial guess (warm start) for solving the perturbed problem and also determines the appropriate algorithm to call for solving the perturbed problem. The arc length or true anomaly angle spanned by the transfer trajectory is the parameter that governs the automated selection of the appropriate perturbed algorithm, and it is based on the respective algorithm convergence characteristics. The second algorithm solves the perturbed problem using the

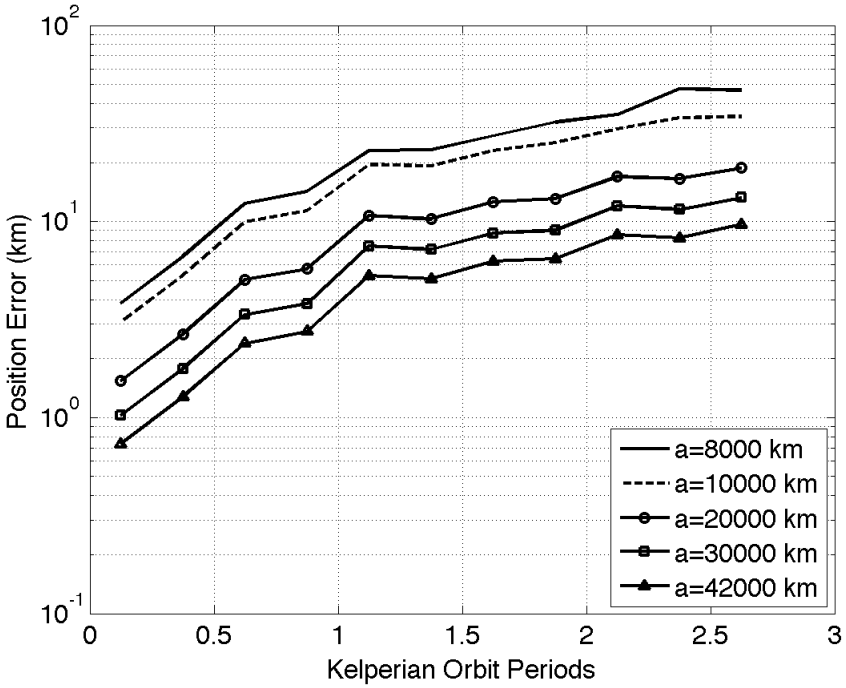


Fig. 16 Position error between the perturbed and Keplerian Lambert solution for five test cases with with zero eccentricity, 10° **inclination and the range of semimajor axis values shown in the legend.**

modified Chebyshev-Picard iteration two-point boundary value algorithm. This algorithm does not require a Newton-like shooting method and is the most efficient of the perturbed solvers presented in the paper, however the domain of convergence is limited to about 1/3 of an orbit (varies with varying eccentricity). The third algorithm extends the domain of convergence of the modified Chebyshev-Picard iteration two-point boundary value solver to about 90% of an orbit, through regularization with the Kustaanheimo-Stiefel transformation. This is the second most efficient of the perturbed set of algorithms. The fourth algorithm uses the method of particular solutions and the modified Chebyshev-Picard iteration initial value algorithm for solving multiple-revolution perturbed transfers. This method does require "shooting" but differs from Newton-like shooting methods in that it does not require propagation of a state transition matrix. The unified Lambert tool makes use of the General Mission Analysis Tool and is capable of computing thousands of perturbed Lambert trajectories in parallel on the Space Situational Awareness computer cluster at the LASR Lab, Texas A&M University. We have demonstrated the power of our tool by solving

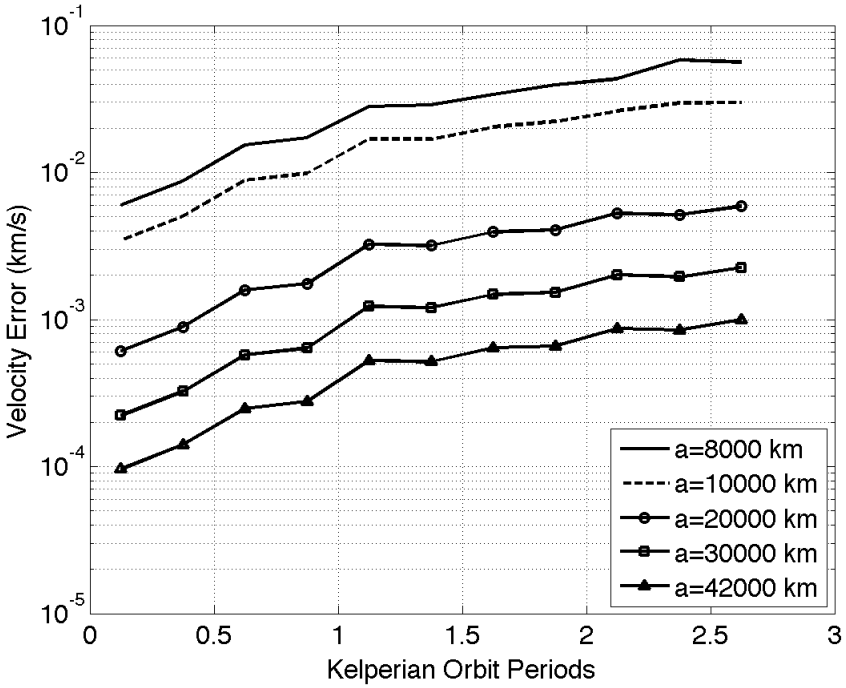


Fig. 17 Velocity error between the perturbed and Keplerian Lambert solution for five test cases with zero eccentricity, 10° inclination and the range of semimajor axis values shown in the legend.

a highly parallel example problem, the generation of extremal field maps for optimal spacecraft rendezvous (and eventual orbit debris removal). In addition we demonstrated the need for including perturbative effects in simulations for satellite tracking or data association. The unified Lambert tool developed in this paper is being used by several industrial partners and we are confident that it will play a significant role in practical applications, including solution of Lambert problems that arise in the current applications focused on enhanced space situational awareness.

Appendix A: Modified Chebyshev-Picard Iteration

Modified Chebyshev Picard Iteration differs from the well-known step integrators such as Gauss-Jackson, Runge-Kutta-Nystrom and ODE45 in that it is a *path approximation* numerical integrator rather than a *step-by-step* integrator. Long state trajectory arcs are approximated continuously and updated at all time instances on each iteration.

The MCPI technique combines Picard iteration with the orthogonal Chebyshev polynomials.

Emile Picard observed that any first order differential equation

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{f}(t, \boldsymbol{x}(t)), \quad t \,\epsilon\, [t_0, t_f], \tag{1}$$

with an initial condition $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ and any integrable right hand side may be rearranged, without approximation, to obtain the following integral equation:

$$\boldsymbol{x}(t) = \boldsymbol{x}(t_0) + \int_{t_0}^{t} f(\tau, \boldsymbol{x}(\tau))d\tau. \tag{2}$$

A sequence of approximate solutions $\boldsymbol{x}^i(t)$, $(i = 1, 2, 3..., \infty)$, of the true solution $\boldsymbol{x}(t)$ that satisfies this differential equation may be obtained through Picard iteration using the following Picard sequence of approximate paths $\left\{\boldsymbol{x}^0(t), \boldsymbol{x}^1(t), ..., \boldsymbol{x}^{i-1}(t), \boldsymbol{x}^i(t), ...\right\}$:

$$\boldsymbol{x}^i(t) = \boldsymbol{x}(t_0) + \int_{t_0}^{t} f(\tau, \boldsymbol{x}^{i-1}(\tau))d\tau, \quad i = 1, 2, ... \tag{3}$$

Picard proved that for smooth, differentiable, single-valued nonlinear functions $\boldsymbol{f}(t, \boldsymbol{x}(t))$, there is a time interval $|t_f - t_0| < \delta$ and a starting trajectory $\boldsymbol{x}^0(t)$ satisfying $\left\|\boldsymbol{x}^0(t) - \boldsymbol{x}(t)\right\|_\infty < \infty$, that for suitable finite bounds (δ, Δ), the Picard sequence of trajectories represents a contraction operator that converges to the unique solution of the initial value problem. The rate of convergence is typically geometric. The guaranteed convergence property sets this method apart from other approaches. The numerical accuracy and efficiency are dominated by the particular process used to carry out the integral; note since the previous $(i-1)$ trajectory approximation is known, the integrand is considered a function of time only. Chebyshev polynomials are used for approximating the integrand in the Picard iteration sequence, and these orthogonal polynomials integrate to produce a Chebyshev series for the integral, including the imposition of initial (or final) boundary conditions.

Appendix B: Modified Chebyshev-Picard Iteration Boundary Value Problem

The formulation presented here is similar to the MCPI-TPBVP algorithm by Bai [7, 14], in that the position trajectory is computed from coefficients where the first two coefficients in the series (involving the unknown velocity constant of integration) are constructed in a way that bypasses this and instead make use of known the initial and final position boundary conditions. However, we present a new formulation for the coefficients that differs from Bai's formulation (although

mathematically equivalent) and is attractive in the sense that it can be constructed in a sparse matrix for easy coding.

A unique characteristic of the MCPI-TPBVP, both Bai's and ours, is that entire series of position coefficients does not require a priori knowledge of the initial velocity, thus allowing TPBVPs to be solved without using a Newton-type shooting method. The MCPI-TPBVP is convergent over about 1/3 of an orbit and it may be extended to about 90% of an orbit with regularization [10]. To solve the Lambert problem over larger arcs a shooting method such as the method of particular solutions [4, 11]) must be used.

The MCPI-TPBVP method may be written mathematically as follows:

$$\boldsymbol{x}^i(\tau) = \sum_{k=0}^{N} \boldsymbol{\alpha}_k^i T_k(\tau) = \boldsymbol{x}(-1) + \int_{-1}^{\tau} \sum_{k=0}^{N-1} \left\{ \boldsymbol{v}(-1) + \int_{-1}^{s} \sum_{k=0}^{N-2} \boldsymbol{F}_k^{i-1} T_k(q) dq \right\} ds, \tag{4}$$

where $\boldsymbol{x}$ is the position trajectory that we wish to compute, $\boldsymbol{\alpha}$ are the coefficients of the position trajectory, T are the Chebyshev basis functions, $\boldsymbol{x(-1)}$ and $\boldsymbol{v(-1)}$ are the position and velocity at the initial time, and $\boldsymbol{F}$ are the coefficients of the acceleration which are determined using least squares approximation. Note that for the first integration the upper limit of the summation for the coefficients approximating the forcing function is $N-2$, and in the second integration the upper limit of the summation is increased to $N-1$. The upper limits are applied because integration increases the degree of the polynomial by one. The approximation of the solution trajectory on the left hand side is of course summed to N. Further details may be found in Woollands' PhD dissertation [8].

The MCPI-TPBVP may be formulated as a "cascade" problem. That is the first integration allows the velocity coefficients ($\boldsymbol{\beta}$) to be obtained in terms of the acceleration coefficients ($\boldsymbol{F}$), and the second integration allows the position coefficients ($\boldsymbol{\alpha}$) to be obtained in terms of the velocity coefficients ($\boldsymbol{\beta}$). This is an elegant approach as there is no intermediate fitting of the velocity that would introduce errors and depart from kinematic consistency.

A. Velocity Approximation

Consider the following velocity approximation for $N = 6$,

$$\boldsymbol{v}^i(\tau) = \sum_{k=0}^{N-1} \boldsymbol{\beta}_k^i T_k(\tau) = \boldsymbol{v}(-1) + \int_{-1}^{\tau} \sum_{k=0}^{N-2} \boldsymbol{F}_k^{i-1} T_k(s) ds. \tag{5}$$

Substituting in the Chebyshev polynomials, simplifying and equating the coefficients of τ leads to the following set of equations:

$$\boldsymbol{\beta}_5^i = \frac{1}{10} \boldsymbol{F}_4^{i-1}, \tag{6}$$

$$\boldsymbol{\beta}_4^i = \frac{1}{8} \boldsymbol{F}_3^{i-1}, \tag{7}$$

$$\boldsymbol{\beta}_3^i = \frac{1}{6} \left(\boldsymbol{F}_2^{i-1} - \boldsymbol{F}_4^{i-1} \right), \tag{8}$$

$$\boldsymbol{\beta}_2^i = \frac{1}{4} \left(\boldsymbol{F}_1^{i-1} - \boldsymbol{F}_3^{i-1} \right), \tag{9}$$

$$\boldsymbol{\beta}_1^i = \frac{1}{2} \left(2\boldsymbol{F}_0^{i-1} - \boldsymbol{F}_2^{i-1} \right), \tag{10}$$

$$\boldsymbol{\beta}_0^i = \boldsymbol{v}(-1) + \left(\boldsymbol{\beta}_1^i - \boldsymbol{\beta}_2^i + \boldsymbol{\beta}_3^i - \boldsymbol{\beta}_4^i + \boldsymbol{\beta}_5^i \right). \tag{11}$$

These can be constructed as a set of general formulae as follows:

$$\boldsymbol{\beta}_0^i = \boldsymbol{v}_0 + \sum_{k=1}^{k=N-1} (-1)^{k+1} \boldsymbol{\beta}_k^i, \tag{12}$$

$$\boldsymbol{\beta}_1^i = \frac{1}{2} \left(2\boldsymbol{F}_0^{i-1} - \boldsymbol{F}_2^{i-1} \right), \tag{13}$$

$$\boldsymbol{\beta}_k^i = \frac{1}{2k} \left(\boldsymbol{F}_{k-1}^{i-1} - \boldsymbol{F}_{k+1}^{i-1} \right), \; k = 1, 2, ..., N-3, \tag{14}$$

$$\boldsymbol{\beta}_{N-2}^i = \frac{\boldsymbol{F}_{N-3}^{i-1}}{2(N-2)}, \tag{15}$$

and

$$\boldsymbol{\beta}_{N-1}^i = \frac{\boldsymbol{F}_{N-2}^{i-1}}{2(N-1)}. \tag{16}$$

B. Position Approximation

Consider the following position approximation for $N = 6$,

$$\boldsymbol{x}^i(\tau) = \sum_{k=0}^{N} \boldsymbol{\alpha}_k^i T_k(\tau) = \boldsymbol{x}(-1) + \int_{-1}^{\tau} \sum_{k=0}^{N-1} \boldsymbol{\beta}_k^i T_k(s) ds. \tag{17}$$

Substituting in the Chebyshev polynomials, simplifying and equating the coefficients of τ leads to the following set of equations:

$$\boldsymbol{\alpha}_6^i = \frac{1}{12}\boldsymbol{\beta}_5^i \tag{18}$$

$$\boldsymbol{\alpha}_5^i = \frac{1}{10}\boldsymbol{\beta}_4^i \tag{19}$$

$$\boldsymbol{\alpha}_4^i = \frac{1}{8}\left(\boldsymbol{\beta}_3^i - \boldsymbol{\beta}_5^i\right) \tag{20}$$

$$\boldsymbol{\alpha}_3^i = \frac{1}{6}\left(\boldsymbol{\beta}_2^i - \boldsymbol{\beta}_4^i\right) \tag{21}$$

$$\boldsymbol{\alpha}_2^i = \frac{1}{4}\left(\boldsymbol{\beta}_1^i - \boldsymbol{\beta}_3^i\right) \tag{22}$$

$$\boldsymbol{\alpha}_1^i = \frac{1}{2}\left(2\boldsymbol{\beta}_0^i - \boldsymbol{\beta}_2^i\right) \tag{23}$$

$$\boldsymbol{\alpha}_0^i = \boldsymbol{x}_0 + \left(\boldsymbol{\alpha}_1^i - \boldsymbol{\alpha}_2^i + \boldsymbol{\alpha}_3^i - \boldsymbol{\alpha}_4^i + \boldsymbol{\alpha}_5^i - \boldsymbol{\alpha}_6^i\right) \tag{24}$$

These can be constructed as a set of general formulae as follows:

$$\boldsymbol{\alpha}_0^i = \boldsymbol{x}_0 + \sum_{k=1}^{k=N} (-1)^{k+1} \boldsymbol{\alpha}_k^i, \tag{25}$$

$$\boldsymbol{\alpha}_1^i = \frac{1}{2}\left(2\boldsymbol{\beta}_0^i - \boldsymbol{\beta}_2^i\right), \tag{26}$$

$$\boldsymbol{\alpha}_k^i = \frac{1}{2k}\left(\boldsymbol{\beta}_{k-1}^i - \boldsymbol{\beta}_{k+1}^i\right),\ k = 1, 2, ..., N-1, \tag{27}$$

$$\boldsymbol{\alpha}_{N-1}^i = \frac{\boldsymbol{\beta}_{N-2}^i}{2(N-1)}, \tag{28}$$

and

$$\boldsymbol{\alpha}_N^i = \frac{\boldsymbol{\beta}_{N-1}^i}{N}. \tag{29}$$

Solving for $\boldsymbol{\alpha}$ directly in terms of $\boldsymbol{F}$ is possible but it leads to a set of complicated formulae that do not exhibit any obvious pattern, thus making it very challenging to code the method for arbitrary N. Instead, formulating the coefficients of $\boldsymbol{\beta}$ in terms of $\boldsymbol{F}$, and then the coefficients of $\boldsymbol{\alpha}$ in terms of $\boldsymbol{\beta}$ as we have done above leads to two nice sets of general formulae that are easily programmable. Note however, that the unknown initial velocity appears in the coefficients of $\boldsymbol{\alpha_0}$ and $\boldsymbol{\alpha_1}$. These two coefficients can be reformulated in terms of the initial and final position vectors thus bypassing the unknown initial velocity.

The first two position coefficients ($\boldsymbol{\alpha_0}$ and $\boldsymbol{\alpha_1}$) are formulated in terms known initial and final position as follows:

$$\boldsymbol{x}(-1) = \sum_{k=0}^{k=N} \boldsymbol{\alpha_k} T_k(-1), \tag{30}$$

$$\boldsymbol{x}(1) = \sum_{k=0}^{k=N} \boldsymbol{\alpha_k} T_k(1), \tag{31}$$

which leads to

$$\boldsymbol{\alpha_0} - \boldsymbol{\alpha_1} + \boldsymbol{\alpha_2} + ... + (-1)^N \boldsymbol{\alpha_N} = \boldsymbol{x}(-1), \tag{32}$$

$$\boldsymbol{\alpha_0} + \boldsymbol{\alpha_1} + \boldsymbol{\alpha_2} + ... + \boldsymbol{\alpha_N} = \boldsymbol{x}(1). \tag{33}$$

This gives two equations and two unknowns thus allowing the first two position coefficients to be recovered as shown:

$$\boldsymbol{\alpha_0} = \boldsymbol{x}(1) + \boldsymbol{x}(-1) - (\boldsymbol{\alpha_2} + \boldsymbol{\alpha_4} + \boldsymbol{\alpha_6} + ...) \tag{34}$$

$$\boldsymbol{\alpha_1} = \boldsymbol{x}(1) - \boldsymbol{x}(-1) - (\boldsymbol{\alpha_3} + \boldsymbol{\alpha_5} + \boldsymbol{\alpha_7} + ...) \tag{35}$$

The position solution is obtained by multiplying the position coefficients ($\boldsymbol{\alpha}$) by the Chebyshev basis functions, where the values of τ are selected by the user in order to provide the solution at desired discrete locations. A "pseudo" velocity can be determined from the current velocity coefficients ($\boldsymbol{\beta}$) in a similar manner, however it will be offset by the unknown initial velocity constant. This constant must first be solved for from the position solution. The procedure for doing this is to

equate the two $\boldsymbol{\alpha_0}$ coefficients shown in Eq. 32 and Eq. 34. The first one is essentially the "IVP" coefficient and the second is the "BVP" coefficient. Equating these clearly reveals that the unknown initial velocity appears linearly and thus can be easily computed. Once this coefficient is included in the velocity Chebyshev series ($\boldsymbol{\beta}$) the true velocity may be obtained.

The matrix representation of the MCPI-TPBVP is shown in two steps. First the "pseudo" velocity coefficients are computed as

$$\boldsymbol{\beta}^i = C_{I1} C_f \boldsymbol{g}(\boldsymbol{X}^{i-1}, \boldsymbol{V}^{i-1}), \tag{36}$$

where $\boldsymbol{g}(\boldsymbol{X}^{i-1}, \boldsymbol{V}^{i-1})$ is the forcing function (acceleration), C_f is the "Chebyshev fit matrix", and C_{I1} is a sparsely populated "integration matrix". The position coefficients are computed as

$$\boldsymbol{\alpha}^i = C_{B2} C_{I1} C_f \boldsymbol{g}(\boldsymbol{X}^{i-1}, \boldsymbol{V}^{i-1}) + \boldsymbol{X}_{0f}. \tag{37}$$

C_{B2} is the sparsely populated integration matrix. $\boldsymbol{X}_{0f}$ is a vector with the first two components being $\boldsymbol{x}_f + \boldsymbol{x}_0$ and $\boldsymbol{x}_f - \boldsymbol{x}_0$ respectively. The remaining elements in this vector are zeros. The position solution is computed as $\boldsymbol{X^i} = C_x \boldsymbol{\alpha}^i$, and the velocity solution is computed as $\boldsymbol{V^i} = C_x \boldsymbol{\beta}^i + C_x \boldsymbol{V}_0$. The "pseudo velocity" $C_x \boldsymbol{\beta}^i$ is the integral of acceleration and differs from the velocity only through the velocity initial condition. $\boldsymbol{V}_0$ is a vector where the first element is $\boldsymbol{v}_0$ and the remaining elements are zeros. $\boldsymbol{v}_0$ is obtained by equating the $\boldsymbol{\alpha}_0^i$ coefficients as described above. For more details on the MCPI-IVP and MCPI-TPBVP refer to Woollands' PhD dissertation [8].

Acknowledgments

We thank our sponsors: AFOSR (Julie Moses) and AFRL (Alok Das, et al) for their support and collaborations under various contracts and grants. We are also grateful to John Prussing for shedding light on the non-uniqueness of solutions to the multiple revolution Lambert problem.

References

[1] R. Battin, *An Introduction to the Mathematics and Methods of Astrodynamics.* AIAA Education Series, 1999.

[2] R. Battin, *Astronautical Guidance.* Mc Graw Hill, 1964.

[3] S. Ochoa and J. Prussing, "Multiple Revolution Solutions to Lambert's Problem," *AAS/AIAA Spaceflight Mechanics Meeting*, 1992.

[4] A. Miele and R. Iyer, "General Technique for Solving Nonlinear, Two-Point Boundary value Problems Via the Method of Particular Solutions," *Journal of Optimization Theory and Applications*, Vol. 5, No. 5, 1970, pp. 392–399.

[5] P. Hack, K. Carbaugh, and K. Simon, "Automated Space Surveillance using the AN/FSY-3 Space Fence System," *Advanced maui Optical and Space Surveillance Technologies Conference*, 2016.

[6] NASA, "Orbital Debris," *Quarterly News*, 2017.

[7] X. Bai, "Modified Chebyshev-Picard Iteration for Solution of Initial Value and Boundary Value Problems," *PhD. Dissertation, Texas A&M, College Station, Texas, USA*, 2010.

[8] R. Woollands, "Regularization and Computational Methods for Precise Solution of Perturbed Orbit Transfer Problems," *PhD. Dissertation, Texas A&M, College Station, Texas, USA*, 2016.

[9] P. Kustaanheimo and E. Stiefel, "Perturbation theory of Kepler motion based on spinor regularzation," *Journal fur die Reine und Angewandt Mathematik*, Vol. 218, 1965, pp. 204–219.

[10] R. Woollands, A. Bani Younes, and J. Junkins, "New Solutions for the Perturbed Lambert Problem Using Regularization and Picard Iteration," *Journal of Guidance, Control and Dynamics*, Vol. 38, 2015, pp. 1548–1562.

[11] R. Woollands, J. Read, A. Probe, and J. Junkins, "Multiple Revolution Solutions for the Perturbed Lambert Problem using the Method of Particular Solutions and Picard Iteration," *Journal of Astronautical Sciences*, accepted 2016.

[12] G. Peterson, "Target Identification and Delta-V Sizing for Active Debris Removal and Improved Tracking Campaigns," *Proceedings of the 23rd International Symposium on Spaceflight Dynamics*, 2012.

[13] D. Vallado, *Fundamentals of Astrodynamics and Applications*. Space Technology Library, 3rd ed., 2007.

[14] X. Bai and J. Junkins, "Modified Chebyshev-Picard Iteration Methods for Solution of Boundary Value Problems," *Advances in the Astronautical Sciences*, Vol. 140, 2011, pp. 381–400.