GTOC9: Results from Team Michigan Tech University-University of Michigan

Ehsan Taheri*, Di Wu, Chi-En Lee, Jared Shimoun
Department of Aerospace Engineering
University of Michigan, Ann Arbor, MI 48109 (USA)

Ossama Abdelkhalik†, Shangyan Zou, Shadi Darani, Brandon Jackson, Jacob Liimatta
Department of Mechanical Engineering-Engineering Mechanics
Michigan Tech University, Houghton, MI 49931 (USA)

May 9, 2017

Abstract

This paper presents the methods developed by the MTU-UoM team in the 9^{th} GTOC along with the obtained results. Several concepts were investigated, specially regarding the selection of the sequence of debris to be removed in each mission. These concepts will be briefed in this paper and the concept that produced this team's best solution is presented in some detail. Overall, genetic algorithms is used as an outer loop optimization tool for determining the sequence of debris to be removed. An inner loop optimizer is used to tune the individual transfers in each mission. This team's best solution consists of 16 missions that removes 122 debris with a cost of 1192.74 MEURs.

*UM team lead. E-mail: etaheri@umich.edu
†MTU team lead. E-mail: ooabdelk@mtu.edu

1 Introduction

The GTOC9 problem description is detailed in reference [5], and it is not presented here to avoid duplication with other papers. The overall goal is to remove 123 debris in a Low-earth orbit (LEO); to avoid the Kessler effect [6]. Some of the orbital elements of the debris are very close to each other, whereas the introduction of the J_2 perturbation changes the right ascension of the ascending node (RAAN) and argument of perigee of the orbits (see Appendix).

For ideal Kepler orbits, it is possible to use the well known Lambert solver to compute the impulsive maneuver needed to rendezvous with a debris, given the initial position of the spacecraft, the final rendezvous position, and the time of flight of the maneuver. Due to the J_2 effect included in this competition, this tool cannot be used to compute an exact trans-

fer. As a result, this team has developed a modified Lambert solver during this competition that results in solutions that are closer to the exact solution compared to the two-body Lambert solver. As a result this modified Lambert solver enabled the optimizer to find better solutions. This modified Lambert approach is described in the Appendix. The search for the best combination of missions resulting in the lowest value of the cost function was performed in two main steps. The first step is a global search among the 123 debris to generate individual missions in a sequential manner. The number of the remaining debris gets smaller as more missions are constructed. Each mission consists of a number of legs; each leg defines a trajectory between two debris. Few different strategies were investigated in this missions construction step.

In the strategy used in the submitted solution, the launch date and flight times of all legs, for each mission, are the design variables of an optimization problem, in which the goal was to minimize the the total impulse value. Later the goal was to find the most efficient missions defined by the ratio of the number of visited debris to the consumed propellant. The optimization problem exploits solutions to multi-revolution modified Lambert problem.

The second step involves a local optimization which is performed over the missions obtained in the first step. Assuming a fixed sequence of debris, the design variables of the local optimization are the launch date and the flight times of all legs of an individual mission.

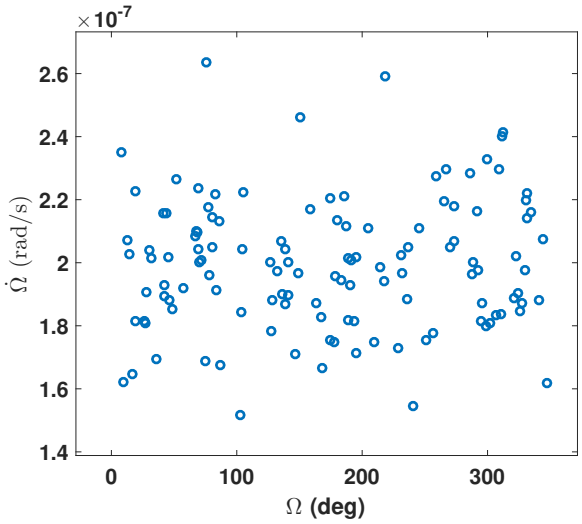

Figure 1: Plot of $\dot{\Omega}$ versus Ω for the respective epoch time of each debri.

2 Sequence of Debris

Given the large number of permutations of the sequence of debris (large design space), it is vital to exploit very rapid measures for conducting a broad search to find the sequence of debris in each mission. Several concepts were investigated; here the most significant of them are briefed and the concept used to generate this team's best solution is detailed at the end.

Semi-Free Rides

Generally speaking, plane-change maneuvers are more expensive compared to in-plane maneuvers. Two angles determine the plane: the inclination and the RAAN. The inclinations of the debris do not change due to the debris motion while the RAAN changes due to the J_2 effect. So, the concept presented in this section utilizes this change in the RAAN due to the J_2 effect, to schedule rendezvous times when the values of RAANs of two debris are very close;

hence the spacecraft can wait with one debris until a good time (when another debris has the same RAAN) and then start a maneuver to rendezvous with the new debris. The spacecraft will then wait with the new debris until a new debris achieves a RAAN that is close to the spacecraft's RAAN, and so on.

This concept generated a nice sequence of debris with very low cost maneuvers; however the time of flight in each maneuver is significantly high violating the 30 days constraint defined in the problem description. As a result the sequences generated using this approach were not submitted.

The planes of all the Debris are changing over time. We can compute the date at which two arbitrary planes would have equal RAAN as follows:

$$\Omega_1 = \Omega_{10} + \dot{\Omega}_1(T - T_{epo1}), \quad (1)$$

$$\Omega_2 = \Omega_{20} + \dot{\Omega}_2(T - T_{epo2}), \quad (2)$$

where T_{epoi} is the epoch date of debris i and T is the current time. If we solve for $\Omega_1 = \Omega_2 + 2n\pi$, we will have:

$$T = \frac{\Omega_{20} - \Omega_{10} + \dot{\Omega}_1 T_{epo1} - \dot{\Omega}_2 T_{epo2} + 2n\pi}{\dot{\Omega}_1 - \dot{\Omega}_2}. \quad (3)$$

The feasible range for the MJD of this problem is from 23467 to 26419. For the cases when the date of the intersection is out of the feasible range, the data is overwritten by a big number.

Although we have the best date to maneuver between the two planes, we still cannot guarantee the spacecraft is exactly at the intersection point. In this work, it is decided that a plane-change maneuver will only be conducted when the spacecraft is close to the intersection point of the two planes so that the cost of the maneuver is lower. So then we compute the time of wait unit the spacecraft can reach the intersection point. The argument of latitude of the intersection and the argument of the spacecraft are computed:

$$A_{sc} = \omega_{sc} + \theta_{sc}, \quad (4)$$

$$\delta\Omega = \Omega_f - \Omega_i, \quad (5)$$

$$\cos\alpha = \cos i_i \cos i_f + \sin i_i \sin i_f \cos\delta\Omega, \quad (6)$$

$$\sin A_{la} = \sin i_f \sin\delta\Omega / \sin\alpha, \quad (7)$$

where A_{sc} is the argument of the position of the spacecraft, the A_{la} is the argument of latitude of the intersection. Ω_f and Ω_i are the right ascension of the initial plane and the final plane, respectively. i_i and i_f are the inclination of the initial plane and the final plane. α is the angle of the plane change. When we find $A_{la} = A_{sc}$ we can start the maneuver. First a single-impulse plane change maneuver is computed. Then an in-plane maneuver is computed to rendezvous the spacecraft with the debris. The combination of the previous two maneuvers is a two-impulse maneuver that will rendezvous the spacecraft with the debris. The cost of the plane-change maneuver can be computed as:

$$\Delta V = 2V_i \sin\alpha/2, \quad (8)$$

where V_i is the velocity of the spacecraft on the initial orbit. The cost of the in-plane transfer can be computed by solving Lambert problem. To have an initial guess for the real cost of the in-plane transfer, we will hold the plane still, and assume the spacecraft has the unperturbed Keplerian motion. Finally, the plane-change maneuver and the departure cost of the in-plane transfer will be combined. This combined maneuver can also be obtained using the

modified Lambert algorithm presented in the Appendix.

Hidden Genes Genetic Algorithms

In this approach, a hidden genes genetic algorithm (HGGA) was implemented to carry out a global search as opposed to a sequential search. Details of the HGGA can be found in [2, 3, 1].

To solve the problem, it can be divided into several missions (m_i) and in each mission some debris (N_{di}) can be captured by a spacecraft. In general, m_i and N_{di} are not known a priori. If we assume that each mission is solved at a time, the only variable that makes the problem a Variable Size Design Space (VSDS) problem is the number of debris at each mission. The design variables in each mission are launch time, arrival time, number of debris (N_{di}), debris IDs (debris that are captured in the mission), wait time, and Deep Space Maneuvers (DSMs) direction and magnitude.

The problem is solved in two phases. In the first phase, the J_2 effect is ignored and launch/arrival time, time of flight, number of debris, and debris IDs are optimized, and in the second step, the effect of J_2 is corrected by adding a DSM in each leg. It is assumed that there is no DSM in the first phase and there is only one DSM in each leg in the second phase. In the first phase, the Lambert problem is solved to find the trajectory between each two debris.

Assume that the current debris is D_i and the next debris is D_{i+1}. At the end of the wait time at D_i and before the departure impulse, the position of the spacecraft is known (similar to the position of debris D_i). Since the time of flight is known, the Lambert problem can be solved to find the departure and arrival impulses. This can be done for all the legs until the last debris of the mission. After the first phase, the effect of the J_2 is corrected by assuming a DSM in each leg. In this algorithm, debris selection is done automatically and there is no need to categorize them into groups. The solution generated using HGGA was not competitive due to the large design space that the HGGA needs to work with.

Sequential Search

Our broad search strategy uses the remaining fuel, where trajectories are built in a sequential manner by adding new legs. The maneuvers are already impulsive for which Lambert problem with a bi-impulsive transfer is considered in the preliminary phase. In the final step, the missions are optimized individually using a local optimizer while taking into account the J_2 perturbation into the governing equations of the spacecraft motion.

Some of the orbital elements of the debris are very close to each other, whereas the introduction of the J_2 perturbation varies the right ascension of the ascending node (RAAN) and argument of perigee of the orbits (see Appendix). Therefore, the debris have different values for $\dot{\Omega}$.

Figure 1 depicts the distribution of the data points on the $\dot{\Omega} - \Omega$ plot. A more important factor, though, is the evolution of the of the RAAN over certain time intervals for it is possible to glean closeness information (in terms of RAAN) in order to form clusters of debris. The evolution of the RAAN is a linear relation,

$$\Omega = \Omega_0 + \dot{\Omega} \times (t - t_0), \tag{9}$$

where Ω_0 is the value of the RAAN at the epoch time (t_0). This linear relation can be

utilized to construct a closeness criterion to be used for clustering debris. On the other hand, the number of debris considered in this problem is significantly smaller than the involved bodies of the previous GTOC problems. While the RAAN has a significant effect on the value of the impulses, initially, we decided to look for generating a series of missions that has the lowest value of cost. Then, we would perform a post-analysis to switch the debris between missions based on the closeness of the RAAN.

Our primary broad search method was to construct individual missions, in a sequential manner, by using a branch-and-prune tree search algorithm. Each mission is built by connecting a series of legs. The building up of sequential legs consists of two main loops: the first loop iterates over the departure debri number and the second loop iterates over the arrival debri number.

To find an optimal solution between each pair of debris, a hybrid optimization method is devised. First, a standard genetic algorithm (GA) performs a broad search over the departure time (MJD_{dep}) and time of flight (TOF) within their defined ranges. The departure time is defined in the range $MJD_{dep} \in [MJD_{LB}, MJD_{UB}]$ where the lower bound and upper bounds of the departure time are $MJD_{LB} = MJD_{arrival} + Stay_{time}$ and $MJD_{UB} = MJD_{LB} + 29$, respectively. The $Stay_{time}$ of 5 days is one of the constraints necessary for deploying the de-orbiting package. In addition, the transfer time between any two debris should not take more than 30 days.

The time of flight of each leg is also defined in the range of $TOF \in [TOF_{LB}, TOF_{UB}]$ where the lower bound and upper bounds of the time of flight are $0.1 \times T_{orb}$ and $5 \times T_{orb}$. Semi-major axis of the debris are relatively close to each other and the period of their orbit is approximately the same. Therefore, we defined the limits of the TOF in terms of the average orbital period T_{orbit} and its value is set to 100 minutes. Two important parameters of a GA are the number of generations and populations. We set those parameters to 20 and 50 respectively.

Eventually, the solution of the GA is used as an initial guess for a local optimizer to further reduce the cost function. For each leg, the departure time (MJD_{dep}) and time of flight (TOF) are the two design variables, and the optimization objective was to minimize the summation of the two impulses,

$$J = \min_{MJD_{dep}, TOF} \Delta V_1 + \Delta V_2, \tag{10}$$

where ΔV_1 and ΔV_2 are the values for impulses at departure and arrival instances, respectively. For the hybrid optimization we did not define any constraint mainly due the fact that the constraint handling is dealt with at the final verification stage in which we use a local optimizer. Each individual execution of GA-Fmincon hybrid optimization takes on average 0.2 seconds (running on 8-cores).

The solution to the Lambert problem is used extensively in the hybrid optimization method, for which we used a multiple-revolution Lambert solver [4] and set the maximum number of revolutions to 5. In addition, we used a compiled Mex file C++ implementation of the Lambert solver. Note that the actual transfer occurs during a short interval. This will simplify the local optimization step during which the accumulative effect of J_2 perturbation becomes small.

Once a solution, which consists of individual missions, was generated through the broad search algorithm, we performed a post-

analysis to modify the missions by performing two major changes. The first change was to remove the last missions that may consist of only one leg, i.e., single-leg missions and to re-assign their debris to the previous missions. The second change was to inspect all of the missions and remove those legs that required significantly greater values of impulse compared to the other legs, and re-assign those debris to other missions.

Debris Re-assignment

The re-assigning strategy that we considered exploits the RAAN closeness which is explained in this section. For any debri which is to be re-assigned, we calculated the closeness criterion

$$\eta = \frac{\sum_i^n (\Omega_{debri-to-assign}(t_i) - \Omega_{debri}(t_i))^2}{n}, \tag{11}$$

where $t_i \in [MJD_{LB}, MJD_{UB}]$. Note that MJD_{LB} and MJD_{UB} correspond to the lower and upper bounds of the mission MJD time interval and are known values. We adopted a simple equi-distant discretization of the mission time interval. n is the number of dicretization points (that depends on the step size used for discretization), $\Omega_{debri-to-assign}$ is the RAAN of the new debri, which is to be re-assigned, evaluated at the descretized points and Ω_{debri} is the RAAN of one of the debri to which we compare the relative differences. The minimum value of the closeness criterion gives us a measure to assign any new debri to a particular mission.

For instance, if there is a mission which already contains 10 debris, we calculate the above parameter by comparing the closeness criterion between the new debri and each of the 10 debris, and take the lowest value. Then, we would repeat the same procedure for the other missions and store the respective RAAN closeness value. Finally, the minimum value of η determines the mission to which we should assign the new debri. The above steps are followed for the other debris until all of them are re-assigned. In addition, we can avoid re-assigning new debris to the original missions from which we picked them. This will ensure that the debris are assigned to new missions.

After performing the above steps, some of the missions will be modified and a new tree-search optimization is performed to achieve a minimum-cost mission that visits all of the debris within each mission. Another consideration is to modify the allowed duration interval of a mission to make sure that there is enough time to visit all of the derbi within each mission. The task of modifying time is the tricky part of assignment. However, the optimization has to be performed over a reduced number of debris within a mission (usually on the order of 25 or less).

3 Results

The final solution consists of 16 missions that deorbits 122 debri with a total cost of 1192.743 MEURs.

Tables 1 to 16 summarizes the individual missions of the submitted solution. Note that each row of the table only reports the dates corresponding to the departure and arrival impulse dates, MJD_{dep} and $MJD_{arrival}$, respectively, between the departure debri number and the arrival debri number. The spacecraft de-orbits a considerable number of debris during the first three missions. Despite our ef-

forts to remove the last two missions and re-assigning their debris into the previous missions, our tree-search algorithm was not capable of finding feasible missions after assigning them to other missions.

Figure 3 depicts the variation of RAAN of the debris visited in the first mission. Note that the apparent separation of bands of lines with similar slope is due to the fact that the angles are not brought into the interval of $[0, 2\pi]$

Table 1: Summary of mission #1

MJD_{dep}	$\text{MJD}_{arrival}$	Dept. Deb #	Arri. Deb #
23680.147	23680.175	19	61
23704.058	23704.303	61	107
23727.367	23727.547	107	30
23736.618	23736.734	30	85
23764.976	23765.083	85	41
23782.362	23782.529	41	45
23789.213	23789.415	45	11
23801.012	23801.203	11	82
23816.460	23816.490	82	71
23844.768	23844.932	71	115
23867.137	23867.386	115	43
23873.892	23874.146	43	47
23900.408	23900.656	47	26
23918.438	23918.679	26	109
23929.680	23929.868	109	7
23958.427	23958.531	7	2

Table 2: Summary of mission #2

MJD_{dep}	$\text{MJD}_{arrival}$	Dept. Deb #	Arri. Deb #
24007.834	24008.027	72	51
24014.289	24014.398	51	10
24020.713	24020.965	10	69
24037.514	24037.657	69	66
24046.992	24047.242	66	73
24074.129	24074.159	73	28
24099.494	24099.639	28	64
24115.494	24115.684	64	52
24144.436	24144.732	52	12
24166.146	24166.399	12	3
24193.416	24193.677	3	31
24202.086	24202.332	31	65
24227.436	24227.702	65	91

Table 3: Summary of mission #3

MJD_{dep}	$\text{MJD}_{arrival}$	Dept. Deb #	Arri. Deb #
24279.950	24279.998	81	13
24288.991	24289.174	13	32
24315.744	24316.037	32	22
24331.734	24331.777	22	17
24352.800	24353.076	17	105
24381.354	24381.452	105	59
24404.094	24404.245	59	98
24426.367	24426.654	98	46
24444.178	24444.216	46	83
24467.346	24467.380	83	48
24495.328	24495.623	48	99
24504.925	24505.101	99	96
24533.435	24533.635	96	114

Table 4: Summary of mission #4

MJD_{dep}	$\text{MJD}_{arrival}$	Dept. Deb #	Arri. Deb #
24593.279	24593.535	0	122
24622.047	24622.337	122	74
24640.378	24640.552	74	119
24667.886	24668.010	119	104
24674.232	24674.284	104	24
24680.834	24681.046	24	108
24707.583	24707.619	108	37

Table 5: Summary of mission #5

MJD_{dep}	$\text{MJD}_{arrival}$	Dept. Deb #	Arri. Deb #
24769.598	24769.714	55	93
24779.413	24779.534	93	100
24801.111	24801.140	100	90
24807.694	24807.994	90	9
24815.556	24815.841	9	33
24842.602	24842.728	33	21
24853.887	24854.082	21	106
24871.127	24871.394	106	68
24893.691	24893.833	68	118
24917.925	24918.101	118	113

Table 6: Summary of mission #6

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
24966.805	24967.058	76	27
24974.934	24975.190	27	20
24996.868	24997.089	20	102
25021.045	25021.332	102	80
25033.293	25033.326	80	121
25062.310	25062.567	121	116
25087.865	25088.165	116	4
25115.532	25115.757	4	15

Table 7: Summary of mission #7

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
25159.145	25159.411	35	1
25167.633	25167.878	1	40
25173.101	25173.358	40	62
25187.643	25187.861	62	54
25212.934	25213.041	54	89
25239.149	25239.255	89	112
25263.523	25263.714	112	87

Table 8: Summary of mission #8

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
25327.371	25327.669	60	103
25341.232	25341.436	103	39
25346.811	25347.072	39	5
25371.826	25372.103	5	53
25377.357	25377.602	53	101
25400.698	25400.955	101	78

Table 9: Summary of mission #9

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
25447.187	25447.477	110	79
25473.209	25473.464	79	34
25485.266	25485.520	34	97
25510.400	25510.575	97	50
25535.869	25536.150	50	86
25542.053	25542.285	86	6

Table 10: Summary of mission #10

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
25584.694	25584.991	25	94
25594.707	25594.828	94	120
25618.420	25618.691	120	38
25623.691	25623.839	38	42
25641.463	25641.636	42	56
25653.755	25653.820	56	111

Table 11: Summary of mission #11

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
25710.022	25710.291	95	8
25734.377	25734.542	8	49
25739.921	25740.060	49	84
25746.989	25747.024	84	36
25769.358	25769.460	36	75

Table 12: Summary of mission #12

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
25871.606	25871.896	88	117
25877.672	25877.972	117	18
25884.791	25885.050	18	70

Table 13: Summary of mission #13

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
25933.550	25933.729	14	58
25962.043	25962.246	58	63

Table 14: Summary of mission #14

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
26098.760	26099.087	57	67
26107.636	26107.911	67	44

Table 15: Summary of mission #15

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
26151.463	26151.788	77	29

Table 16: Summary of mission #16

MJD_{dep}	$MJD_{arrival}$	Dept. Deb #	Arri. Deb #
26205.748	26206.085	23	16

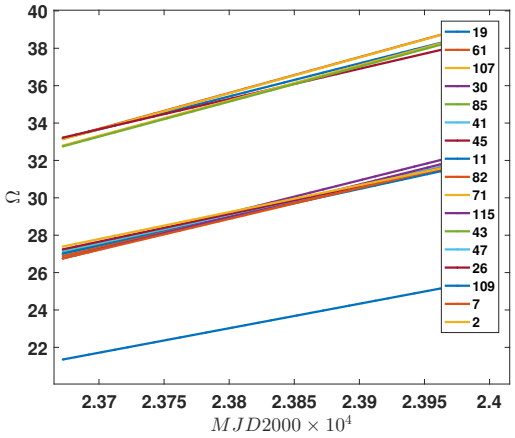

Figure 2: Evolution of RAAN of the debris of the first mission $MJD2000 \in [23672.248, 23963.531]$.

Acknowledgment

This work was partially funded by National Science Foundation, award # 1446622

4 Conclusion

Team MTU-UoM employed a set of tools which were sufficient to find a good solution to GTOC9 problem. A major enhancement would have been to utilize an efficient tree-search algorithm. In addition, it would be ideal to perform, early on, a clustering strategy in terms of the right-ascension of the ascending node, and then focus on visiting the debris within each cluster. In addition, for each mission, plot of $\dot{\Omega} - \Omega$ is helpful in fixing the sequence of debris (transfers) and consider only the stay time and the time of transfer as design variables. The debri re-assignment strategy that we considered during the competition time can be performed more efficiently.

Although we made progress in GTOC9, there is a considerable gap between our solution and the top-rank teams. There are still a lot of works for us to do in trajectory design and optimization. We have only used personal desktop computers and exploited parallel capability of MATLAB running our codes on eight cores. It is reasonable to run our codes on clusters with access to a greater number of cores. We should also consider developing our codes on compiled programming languages, such as C or Fortran. In addition, developing a capable local optimizer (other than MATLAB's fmincon) is quite important for achieving improved solutions.

References

[1] Ossama Abdelkhalik. Hidden genes genetic optimization for variable-size design space problems. *Journal of Optimization Theory and Applications*, 156(2):450–468, 2013.

[2] Ossama Abdelkhalik and Shadi Darani. Hidden genes genetic algorithms for systems architecture optimization. In *Genetic and Evolutionary Computation Conference*, Denever, CO, July, 20–24 2016. Association for Computing Machinery Special Interest Group on Genetic and Evolutionary Computation, Advancing Computing Machinary (ACM).

[3] Ahmed Gad and Ossama Abdelkhalik. Hidden genes genetic algorithm for multi-gravity-assist trajectories optimization. *AIAA Journal of Spacecraft and Rockets*, 48(4):629–641, July?-August 2011.

[4] RH Gooding. A procedure for the solution of lambert's orbital boundary-value prob-

lem. *Celestial Mechanics and Dynamical Astronomy*, 48(2):145–165, 1990.

[5] Dario Izzo. *Problem description for the 9th Global Trajectory Optimisation Competition*, May 2017.

[6] Donald J Kessler and Burton G Cour-Palais. Collision frequency of artificial satellites: The creation of a debris belt. *Journal of Geophysical Research: Space Physics*, 83(A6):2637–2646, 1978.

Appendix: Modified Lambert Solver

The Lambert solver finds the bi-impulse maneuver necessary to rendezvous with a debris given the initial spacecraft position, the final rendezvous position, and the time of flight of the maneuver, assuming Keplerian motion. Due to the oblateness of the Earth, the keplerian motion will be perturbed by the J_2 effect. The J_2 effect will influence the right ascension, the argument of the perigee and the mean anomaly. The latter was neglected in GTOC 9.

$$\dot{\Omega} = -\frac{3}{2} J_2 (\frac{r_{eq}}{p})^2 n \cos i, \tag{12}$$

$$\dot{\omega} = \frac{3}{4} J_2 (\frac{r_{eq}}{p})^2 n (5\cos^2 i - 1). \tag{13}$$

The change rate of Ω and ω is linear.

$$\Omega - \Omega_0 = \dot{\Omega}(t - t_0), \tag{14}$$

$$\omega - \omega_0 = \dot{\omega}(t - t_0). \tag{15}$$

To solve the two-body transfer problem with the J_2 effect, the following strategy is developed. For a two-body transfer problem without J_2 effect and with the date of the departure, the arrival and the fixed time of flight (TOF), usually Lambert problem is used to find the transfer orbit. The required impulse for the departure Δv^d and the arrival Δv^a can be calculated. However, the problem under study takes into account the J_2 effect. Here, the solution obtained from Lambert is used as initial guess. An optimization algorithm will be introduced here for solving the perturbed transfer orbit. The objective function is constructed as:

$$J = \|\vec{r}^a_{sc} - \vec{r}^a_{Debris}\|, \tag{16}$$

where $\vec{r}^a_{sc}$ is the position vector of the spacecraft at the final time which is also the arrival time. $\vec{r}^a_{Debris}$ is the position vector of the debris at the arrival time. The goal of the optimization is to minimize the objective function which means we want to satisfy the rendezvous condition. The variables to be optimized is $\vec{v}^d_{sc}$, the velocity of the spacecraft at departure time. The optimization algorithm is setup as:

$$\begin{aligned}
Min \quad &: \quad J = \|\vec{r}^a_{sc} - \vec{r}^a_{Debris}\|, \\
s.t \quad &: \\
\ddot{x} &= -\frac{\mu x}{r^3}(1 + \frac{3}{2} J_2 (\frac{r_{eq}}{r})^2 (1 - 5\frac{z^2}{r^2})), \\
\ddot{y} &= -\frac{\mu y}{r^3}(1 + \frac{3}{2} J_2 (\frac{r_{eq}}{r})^2 (1 - 5\frac{z^2}{r^2})), \\
\ddot{z} &= -\frac{\mu z}{r^3}(1 + \frac{3}{2} J_2 (\frac{r_{eq}}{r})^2 (3 - 5\frac{z^2}{r^2})), \\
&\|\vec{r}^a_{sc} - \vec{r}^a_{Debris}\| < 0.1,
\end{aligned}$$

where the first three constraints represent the perturbed Keplerian motion. The last constraint is for the rendezvous condition - the difference between the position of the spacecraft and the position of the debris cannot be bigger than 0.1 km. Due to the J_2 effect, if we propagate the perturbed trajectory with

the $\vec{v}_{sc}^{d}$ obtained from original Lambert solution, the final position of the spacecraft does not reach $\vec{r}_{Debris}^{a}$. To take advantage of J_2 effect, the Lambert problem can be used to solve the transfer orbit between $\vec{r}_{sc}^{d}$ and $\vec{r}_{temp}^{a}$. A temporary position $\vec{r}_{temp}^{a}$ is computed from a modified set of orbital elements. Assume the orbital elements at the arrival time are $[a^a,\, e^a,\, i^a,\, \Omega^a,\, \omega^a,\, M^a]^T$. The modified orbital elements are computed from:

$$\Omega_{temp} = \Omega^a - \dot{\Omega}T, \tag{17}$$

$$\omega_{temp} = \omega^a - \dot{\omega}T, \tag{18}$$

where T is the time of flight. So the temporary position is computed from the modified orbital elements: $[a^a,\, e^a,\, i^a,\, \Omega_{temp},\, \omega_{temp},\, M^a]^T$. In this competition, $\dot{\Omega}$ is always a positive number while $\dot{\omega}$ is always negative. If Ω^a is greater than Ω^d which is the right ascension of the orbit before we apply the departure impulse which means we have the possibility to take advantage of J_2 effect. After we compute Ω_{temp}, if we found $\Omega_{temp} > \Omega^d$, then we are able to apply the modification for the orbital elements, otherwise we will apply $\Omega_{temp} = \Omega^d$. If Ω^a is smaller than Ω^d, which means we are moving against J_2 effect, the modification for the orbital elements would not be applied, we will have $\Omega_{temp} > \Omega^a$. Since $\dot{\omega}$ is negative, so the opposite algorithm will be applied to compute ω_{temp}. The description above can be summarized as

if $\Omega^a < \Omega^d$ **then**
  $\Omega_{temp} = \Omega^a$
else
  if $\Omega_{temp} > \Omega^d$ **then**
    $\Omega_{temp} =$ Eq. (17)
  else
    $\Omega_{temp} = \Omega^d$
  end if
end if
if $\omega^a > \omega^d$ **then**
  $\omega_{temp} = \omega^a$
else
  if $\omega_{temp} > \omega^d$ **then**
    $\omega_{temp} = \omega^d$
  else
    $\omega_{temp} =$ Eq. (18)
  end if
end if

So the above logic along with the Eqs. (17) and (18) will be applied to compute the temporary position. The Lambert problem will be solved between the initial position and the temporary position. The solution from Lambert problem will be taken as the initial condition of the optimization. Finally, we can use Eq. ((17)) to optimize the velocity of the spacecraft at the departure point to reach the final position. The solution will be the real cost of the perturbed transfer between two points.