AAS 10-141

OPTIMAL DESPIN OF A TUMBLING SATELLITE WITH AN ARBITRARY THRUSTER CONFIGURATION, INERTIA MATRIX, AND COST FUNCTIONAL

Daniel Sheinfeld[*] and Stephen Rock[†]

An algorithm to calculate the optimal control to despin a tumbling satellite with known, but arbitrary thruster configuration, inertia matrix, and cost functional is presented. While general, the algorithm is applied here to the case of an arbitrary set of impulsive thrusters, arbitrary inertia matrix, and a minimum fuel consumption cost functional. The control is calculated as a set of optimal switching surfaces using numerical dynamic programming to generate optimal state trajectories and control histories. Points along the state trajectories where the control switches are points on the switching surfaces. Alternative state and staging variables provide substantially improved performance of the algorithm. Computation time savings on the order of a factor of *40* and data storage savings on the order of a factor *2000* resulting from the use of the alternative state and staging variables are demonstrated through simulation.

INTRODUCTION

A critical issue that must be addressed to enable on-orbit servicing of satellites is how to capture and despin a satellite that may be tumbling. A satellite can lose attitude control and begin to tumble for any number of reasons: thruster failure, avionics failure, failed docking attempts, collision with space debris, etc. In general, it cannot be assumed that the tumbling motion is benign.

The satellite considered here is modeled as a rigid body with arbitrary, but known a priori or measured, inertia parameters. Disturbance torques are assumed negligible so the satellite is nominally undergoing torque-free motion. The goal is to reduce the angular momentum of the satellite, ***h***, to zero by applying external forces and torques while minimizing some cost functional (fuel, time, etc.). The angular position and location of the center of mass are not considered.

In developing an algorithm to solve this problem, it is assumed that measurements of the angular velocity of the tumbling satellite expressed in a body-fixed frame are available (angular position is not required). Note that sensors which provide these data either exist or are under development. An example is the system implemented by Zampato, et. al.[17] This system tracks features on a rotating rigid body using a stereo camera pair and outputs attitude and attitude rate. Another

[*] PhD Candidate, Department of Aeronautics and Astronautics, Stanford University, Durand Building Room 250, 496 Lomita Mall, Stanford, California 94305, USA.

[†] Professor, Department of Aeronautics and Astronautics, Stanford University, Durand Building Room 250, 496 Lomita Mall, Stanford, California 94305, USA.

example is the algorithm developed by Dubowsky and Lichter which produces a measurement of attitude from a 3D reconstruction of the body using range data.[4]

Various types of actuators can be used to execute the despin process, however the assumption here is that the tumbling satellite is underactuated. If the tumbling satellite has some working thrusters, then they may be used. Alternatively, a small external thruster may be affixed to the tumbling satellite and fired remotely. If the tumbling satellite has a docking ring (such as on the Hubble Space Telescope), a chaser satellite may rigidly dock with the tumbling satellite and then apply forces or torques using thrusters on the chaser. In other proposed schemes, impulsive forces are applied to the tumbling satellite by direct contact with the end-effector of a robotic arm mounted on the chaser or by firing soft projectiles at the tumbling satellite.[5,14,16]

Previous Work

This paper extends the prior work of Ioslovich, et. al.[1,2,7,8,9,10,11] Ioslovich solves the despin problem using a variational technique which seeks to satisfy the Krotov-Bellman optimality conditions.[13] The resulting optimal control is a set of switching surfaces in a three-dimensional state space which has $\boldsymbol{h}$ as the state vector. Ioslovich gives closed-form expressions for the switching surfaces for some specific problem configurations (e.g. a set of three impulsive thrusters aligned with the principle axes, an asymmetric inertia matrix, fuel consumption cost functional). In contrast, the algorithm presented in this paper generates the switching surfaces numerically for a broader set of problem configurations: an arbitrary set of impulsive thrusters, an arbitrary inertia matrix, and a fuel consumption cost functional. This encompasses a wide range of problem configurations that are likely to be encountered. Although the algorithm is specifically tailored for this set of problem configurations, it can also be applied to more general problem configurations with arbitrary thruster configurations, inertia matrices, and cost functionals.

Algorithm Outline

The algorithm presented here has two steps. The first step uses numerical dynamic programming (NDP) to generate optimal state trajectories and control histories (the use of NDP for calculating optimal despin trajectories was suggested by Melton, et. al.).[15] Points along the state trajectories where the control switches are points on the switching surfaces. The second step uses these points to generate the switching surfaces numerically (this method of generating switching surfaces was suggested by Gragg).[6] To improve computational efficiency, the switching surfaces are only generated around points where the state vector is nominally expected to cross the switching surfaces given the initial conditions.

A key issue to address in formulating the despin problem is the choice of state and staging variables for use in the NDP step. This choice has a significant impact on the computation time and data storage requirements of the overall algorithm. For the despin problem, NDP can be implemented using the three components of $\boldsymbol{h}$ as the state variables and time as the staging variable. Here however, NDP is implemented in a manner which incorporates knowledge of the underlying dynamics through an alternative set of state and staging variables defined by the angular momentum loci (these variable choices are inspired by Ioslovich, et. al.).[1,2,7,8,9,10,11]

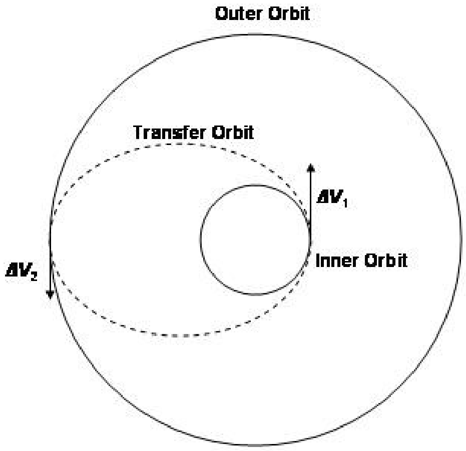

Figure 1. Orbital Transfer Example.

A simple example that illustrates the concept of alternative state and staging variables and their value is the orbit transfer of a satellite. Specifically, a satellite in a circular orbit around the earth is to be transferred to a higher, coplanar circular orbit with minimum fuel expenditure. From orbital mechanics, the form of the optimal control and state trajectory for this problem is known. The satellite is allowed to orbit until the proper initial firing point is reached, the optimal delta-v (from the set of available delta-v values) is applied to place the satellite on a transfer orbit, the satellite is allowed to orbit until the proper final firing point is reached, and a second delta-v is applied to place the satellite on the final desired circular orbit (Figure 1). A standard approach to solving this problem would be to generate an optimal control time history, $\boldsymbol{u}(t)$, and corresponding state trajectory, $\boldsymbol{x}(t)$. However, the same solution can be found with substantially less computation time and data storage by incorporating the known form of the optimal control into the state and staging variables. The current orbit and position along the orbit are the state variables and the orbit is the staging variable. Using these state and staging variables, the problem reduces to figuring out which delta-v to apply and when to apply it in order to jump optimally from one orbit to the next. These choices are the control decisions. In other words, the problem is really finding the optimal control for each orbit and the optimal sequence of orbits rather than finding the optimal control and state at each time.

State Variable Definitions

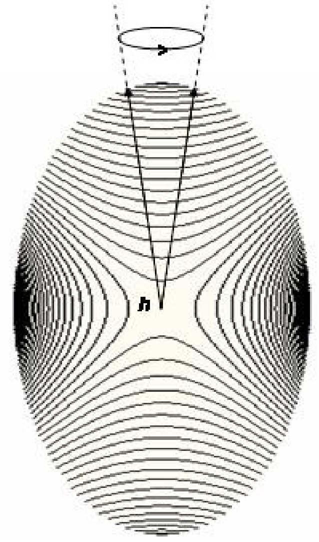

Figure 2. Angular Momentum Loci (Asymmetric Inertia Matrix).

For the despin problem, the alternative set of state variables are two scalar quantities which describe the current angular momentum locus and the position of ***h*** along the current angular momentum locus. The alternative staging variable is the current angular momentum locus.

The angular momentum loci give a simple representation of torque-free motion. The motion of the tumbling satellite is governed by Euler's equations: $\boldsymbol{u} = \left(\dot{\boldsymbol{h}}\right)_f + \boldsymbol{\omega}\times\boldsymbol{h}$, where ***u*** is the applied torque, ***h*** is the angular momentum, ***ω*** is the angular velocity, and $(\dot{\ })_f$ indicates the time derivative as viewed by an observer in the body frame. With zero applied torque, the trajectories of ***h*** in the body frame are closed. These closed trajectories are known as angular momentum loci or simply loci.[3] The loci are similar in shape to the polhodes (which are the closed trajectories traced out by ***ω*** in the body frame, with zero applied torque). As the body tumbles, the tip of ***h*** travels along one of the loci (much like the tip of ***ω*** travels along one of the polhodes). Figure 2 illustrates example sets of possible loci for an asymmetric inertia matrix. Each locus is described mathematically by two scalar quantities: the ratio of $\boldsymbol{h}^T\boldsymbol{h}$ to *2KE* (*KE* = kinetic energy due to rotation) and the magnitude of ***h***. The ratio of $\boldsymbol{h}^T\boldsymbol{h}$ to *2KE* determines the shape of the locus and the magnitude of ***h*** determines the size of the locus.

Utilizing the loci, the despin problem is approached in the same manner as the orbit transfer problem. Again, the despin problem addressed has an arbitrary set of impulsive thrusters, an arbitrary inertia matrix, and a fuel consumption cost functional. Given the similarities between the orbit transfer problem and the despin problem, it is expected that the form of the optimal control for the despin problem is similar in nature to the form of the optimal control for the orbit transfer problem. Calculating the optimal control for the despin problem confirms this expectation. The satellite is allowed to tumble until ***h*** is optimally aligned in the body frame, the optimal delta-h (from the set of available delta-h values) is applied, the satellite is allowed to tumble, a delta-h is applied, the satellite is allowed to tumble, a delta-h is applied, etc., until ***h*** is reduced to zero. Again, this solution may be obtained in a variety of ways, and a standard approach would be to generate an optimal control time history, ***u***(*t*), and corresponding state trajectory, ***x***(*t*). However, the same solution can be found with substantially less computation time and data storage by incorporating the known form of the optimal control into the state and staging variables. Using the alternative state and staging variables, the problem reduces to figuring out which delta-h to apply and when to apply it in order to jump optimally from one locus to the next. These choices are the control decisions. In other words, the problem is really finding the optimal control for each locus and the optimal sequence of loci rather than finding the optimal control and state at each time.

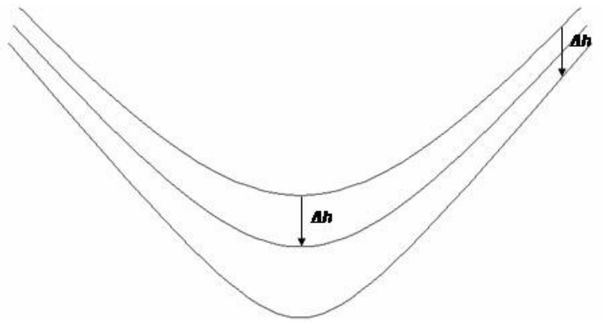

Figure 3. Single Locus Transition.

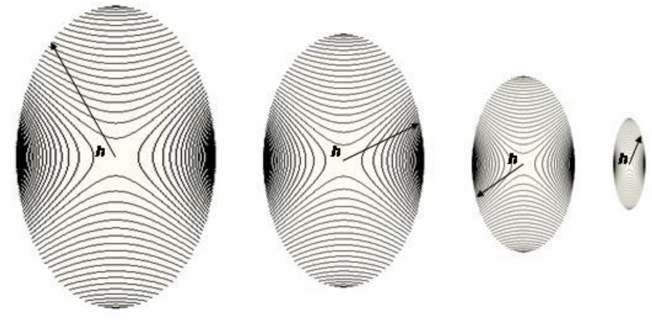

Figure 4. Optimal Locus Transition Sequence.

The concept of jumping from one locus to the next is illustrated through an example. For this example, the set of available control torques is a single impulsive torque (i.e. a delta-h) aligned with a body-fixed direction. Figure 3 depicts two possible locations to apply this delta-h on a particular locus. In this example, the topmost locus is the starting locus, and a transition can be made to either the middle or the bottom locus depending on where the delta-h is applied. The job of the optimization algorithm is to make this choice. As the satellite is despun, a sequence of these choices is made. Figure 4 depicts an optimal locus transition sequence (proceeding from left to right). As the satellite is despun, ***h*** jumps from one locus to the next. The shrinking inertia ellipsoid illustrates that the magnitude of ***h*** is decreasing throughout this particular optimal locus transition sequence (although in general, the magnitude of ***h*** need not decrease monotonically during an optimal locus transition sequence).

The control which implements optimal jumping from one locus to the next may be implemented as a switching surface. As was described above, there is an optimal firing point and an optimal delta-h to apply for each locus. As ***h*** moves along the current locus, it is guaranteed to cross the optimal firing point at some time. The position of ***h*** along the orbit is monitored, and when the optimal firing point is crossed, a delta-h is applied.

ALGORITHM

The algorithm to calculate the optimal control to despin a tumbling satellite has two steps. The first step generates optimal state trajectories and control histories using NDP. The second step numerically generates optimal switching surfaces using the results of the first step. These steps are described below.

Step 1: Numerical Dynamic Programming (NDP)

The description of the NDP algorithm which follows assumes that total fuel expenditure is being minimized. The algorithm may be extended to minimize total time, and this is discussed later.

The first step of generating the switching surfaces is to use NDP to produce optimal state trajectories and control sequences. The NDP algorithm generates a large look-up table which provides the optimal control at every point in the state space for every stage of the control sequence.[12] Generating the NDP look-up table is the most expensive step of the entire surface approximation algorithm (the costs of the rest of the algorithm are a small fraction of the cost of the look-up table). The look-up table is built up one stage at a time. The total cost of the look-up table is equal to the number of stages multiplied by the cost of each stage. The alternative state variables reduce the data storage cost of each stage, and the alternative staging variable reduces the number of stages.

State Variables

In NDP, the state grid defines the points in the state space for which optimal control information is calculated and stored in the look-up table at each stage. With the alternative formulation, all that needs to be stored is the optimal firing point and optimal delta-h for each locus. Thus, the alternative set of state variables allows for significant savings in data storage for each stage. The savings in data storage cost is proportional to the number of points along each locus. If there are n points per locus, the storage cost of each stage is reduced roughly by a factor of n.

Staging Variable

In NDP, the staging variable determines when new control decisions are made. A typical staging variable is time.[12] With this staging variable, control decisions are made at regular dt time intervals which means that new control decisions are made at discrete points along each locus. This results in a "fire, do nothing, do nothing, …, do nothing, fire, do nothing, etc." control sequence because there is only one optimal firing point for each locus. Each "do nothing" is a control decision. Once again, this is inefficient. As was shown above, the only control decisions which need to be made for each locus are which delta-h to apply and the position along the locus at which to apply it. Thus in the alternative formulation, the staging variable is chosen to be the locus itself, and all of the "do nothing" stages are eliminated. The number of stages is reduced by a factor of m where $1 \leq m \leq n$. This value depends on the parameters of the specific problem being solved.

NDP Look-up Table

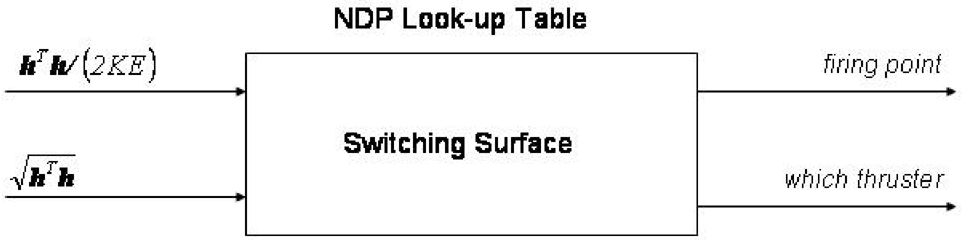
NDP Look-up Table
$\boldsymbol{h}^T\boldsymbol{h}/(2KE)$
$\sqrt{\boldsymbol{h}^T\boldsymbol{h}}$
Switching Surface
firing point
which thruster

Figure 5. NDP Look-up Table.

The output of the NDP process is the look-up table depicted in Figure 5. The state variables which describe the current locus are the inputs to the look-up table. The control variables, which thruster (i.e. which delta-h) to fire and the optimal firing point along the locus, are the outputs of the look-up table. The look-up table is accessed after each locus transition since the current locus is the staging variable. After each locus transition, the current locus is given to the look-up table and the optimal thruster and firing point are returned. The next impulse is applied, a locus transition occurs, and the process repeats. Points along the state trajectories where impulses are applied (i.e. where the control switches) are points on the switching surfaces. Thus, the NDP look-up table provides a numerical approximation to the optimal switching surfaces.

Total Savings

Table 1. Cost Comparison.

Typical	Alternative
state = h_x, h_y, h_z	state = *locus*, *position along locus*
stage = *time*	stage = *locus*
total time = O(m*p)	total time = O(p)
total storage = O(mn*q)	total storage = O(q)

The performance benefits of the alternative formulation are decreased total computation time and total data storage requirement of the look-up table. The total computational time is proportional to the number of stages. The total data storage requirement is equal to the requirement per stage multiplied by the number of stages. The number of points on each locus in the alternative state space, n, determines the performance benefit. The number of stages is reduced by a factor of m where $1 \leq m \leq n$. The value of m is equal to the average number of "do nothing" commands in-between in each "fire" command plus one. The data storage requirement at each stage is approximately reduced by a factor of n. The savings are shown in Table 1.

There is a small amount of extra bookkeeping which is necessary to implement the alternative formulation. When the state space is initially discretized, both state representations of each grid point (the [h_x, h_y, h_z] value and the corresponding locus and position) must be stored. The actual $\boldsymbol{h}$ values are needed when controls are applied in the NDP algorithm. Also, there is an ambiguity in the alternative state representation because there is always a pair of loci with the same ratio of $\boldsymbol{h}^T\boldsymbol{h}$ to *2KE*. A flag needs to be set to distinguish the two loci.

Step 2: Surface Fitting

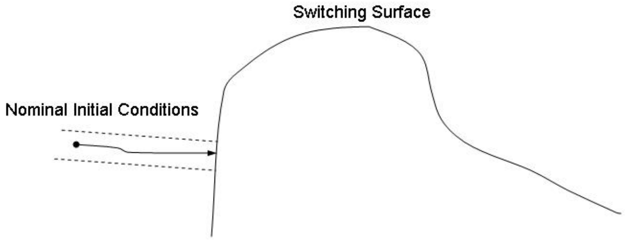

Figure 6. Optimal Switching Surface for a Single Stage.

The second step of the algorithm is to generate numerically the switching surfaces using the optimal state trajectories and control sequences generated with NDP. The points along the optimal state trajectories where impulses are applied (i.e. where the control switches) are points on the switching surfaces. The shape of each surface is dependent upon how many impulses remain to be applied to the target. This is because the optimal control at the current stage depends on the future controls (this is the principle of optimality of dynamic programming).[12] So after each impulse from the optimal control sequence is applied, the switching surface changes shape. Thus, the optimal state trajectories intersect a new switching surface at each impulse. This means that a set of switching surfaces is being interpolated rather than a single surface. Figure 6 shows the optimal switching surface at a single stage.

In order to generate good representations of the switching surfaces, a sufficiently fine state grid and a large number of initial conditions are used. This gives a large number of points on the switching surfaces. These points may be interpolated to generate the switching surfaces. The shapes of the underlying switching surfaces are unknown a priori so a trial and error approach is used to choose the shape.

The entire switching surface at each stage may be interpolated if desired, however for the despin problem this is not necessary because the nominal initial conditions of the target are known. Thus, it is only necessary to calculate optimal state trajectories and control histories for the nominal initial conditions and for a region of initial conditions around the nominal to account for sensor noise and disturbance torques (the region can be sized based on the expected magnitudes of the noise and disturbances). This effectively creates a tunnel through the state space where the optimal trajectories are calculated. With this paradigm, the switching surfaces are interpolated locally. The interpolation is easier to perform locally because the switching surface is simpler. For instance, a locally linear (or planar, etc.) interpolation may be used. This situation is depicted in Figure 6.

This technique works as long as the state remains within the tunnel which is not guaranteed. To deal with this, the tunnel is regenerated using the measured state after each impulse is applied. Thus, each new tunnel centers around the nominal initial conditions after each impulse. The tunnel can also be generated more frequently if necessary.

SIMULATION RESULTS

Results from numerical trials are used to demonstrate the performance benefits of using the alternative state and staging variables versus using the three components of ***h*** and time. Comparisons are made based on the computation and storage of the NDP look-up table.

Each trial involves a satellite with a random inertia matrix, initial angular momentum, and discrete control input set. Each control input set consists of 3 (non-reversible) impulsive torques with random directions and magnitudes. The total number of grid points is selected (same for both formulations), and a grid is laid over both state spaces. The total number of grid points is *150,000* for this set of trials. The look-up table is generated using both methods. Both methods are run until *10* impulses have been applied (only *10* impulses are used for simplicity). Both methods give the same value of minimum cost (the two methods may select different optimal control sequences, if there is more than one). Fuel optimality and time optimality are considered. The data are given in Tables 2 and 3.

Table 2. Simulation Results (Fuel Optimal).

Computation Time (min)	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**	**Trial 5**	**Trial 6**	**Trial 7**
Alternative	*1.16*	*1.16*	*1.16*	*1.16*	*1.16*	*1.16*	*1.16*
Typical	*35.4*	*15.4*	*39.0*	*33.9*	*27.5*	*11.8*	*19.8*
Data Storage (MB)	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**	**Trial 5**	**Trial 6**	**Trial 7**
Alternative	*0.045*	*0.045*	*0.045*	*0.045*	*0.045*	*0.045*	*0.045*
Typical	*94.5*	*40.5*	*104*	*90.0*	*76.5*	*31.9*	*54.0*

Table 2 shows typical trial results given the above parameters for the fuel expenditure cost functional. The alternative formulation is always run for *10* stages such that there are *10* impulses. Thus, the computation time and data storage requirements for the alternative formulation are the same for each trial. The number of stages required to ensure *10* impulses in the other formulation varies however (depending on the value of *m* discussed above). The performance benefit of the alternative formulation is directly related to the number of points on each locus (this value is *100* for all of the trials given in Table 2). Therefore, the more points on each locus, the better the alternative formulation performs relative to the other formulation. This is an attractive feature because increasing the number of points on each locus gives better resolution for calculating the optimal firing point for each locus. The other formulation never performs significantly better than the alternative formulation in terms of computation time and data storage. In the extreme case, there is one point per locus and both methods require approximately the same amount of computation time and data storage.

Time Optimality

The algorithm as described above is tailored for the case of despin with minimum fuel expenditure. However, it may also be modified for the case of despin with minimum total time. With total fuel expenditure as the cost functional, the optimal firing point is calculated for each locus. When a transition to a new locus occurs, the state is allowed to evolve until the optimal firing point is reached. However, with total time as the cost functional, there is a penalty for waiting for a particular firing point, and the starting point along the locus (after a transition) must be considered. Thus, the optimal firing point is calculated and stored for each potential starting point. Because of this, the data storage costs for each stage under both formulations are now equal. I.e., the overall data storage benefit of the alternative formulation has decreased. There is also an increase in computation time cost. For many cases however, there is still an overall benefit (in terms of computation time and data storage) to using the alternative formulation (Table 3).

Table 3. Simulation Results (Time Optimal).

Computation Time (min)	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**	**Trial 5**	**Trial 6**	**Trial 7**
Alternative	*2.15*	*2.15*	*2.15*	*2.15*	*2.15*	*2.15*	*2.15*
Typical	*8.02*	*3.20*	*16.4*	*1.62*	*4.92*	*3.29*	*16.3*
Data Storage (MB)	**Trial 1**	**Trial 2**	**Trial 3**	**Trial 4**	**Trial 5**	**Trial 6**	**Trial 7**
Alternative	*4.5*	*4.5*	*4.5*	*4.5*	*4.5*	*4.5*	*4.5*
Typical	*22.5*	*9*	*45*	*4.5*	*13.5*	*9*	*45*

Table 3 shows that the computation time and data storage benefits of the alternative formulation are decreased for the total time cost functional. This result is due to two factors. The first is that the optimal firing point is now calculated and stored for each point on each locus. This increases the total computation time and data storage cost. The total data storage cost increases by a factor equal to the number of points per locus (*100* for all of the trials given in Table 3). Also when the penalty for waiting is added, the number of "do nothing" commands in between each fire command naturally decreases. And as the weight on total time is increased, the number of "do nothing" commands decreases further. This can lead to the situation (as in Trial 4) where the other formulation is actually superior to the alternative formulation in terms of computation time. In this case, the cost of eliminating the "do nothing" commands outweighs the cost of leaving them in. This situation does not arise when considering total fuel expenditure because there are more "do nothing" commands (typically) and the cost of removing them is insignificant.

CONCLUSION

An algorithm to calculate the optimal control to despin a tumbling satellite with an arbitrary set of impulsive thrusters, an arbitrary inertia matrix, and the fuel consumption cost functional was presented. The algorithm was tailored for the case of an arbitrary set of impulsive thrusters, arbitrary inertia matrix, and a minimum fuel consumption cost functional, but it may be extended to include arbitrary thruster configurations and arbitrary cost functionals. Significant savings in terms of computation time and data storage were achieved through an appropriate choice of state and staging variables.

ACKNOWLEDGEMENTS

Daniel Sheinfeld is supported by a Cleve Moler Stanford Graduate Fellowship, a National Defense Science and Engineering Graduate Fellowship, and The Institute for Dexterous Space Robotics (NASA grant number Z627401-A).

REFERENCES

[1] M. Z. Borshchevskii, et. al., “Optimal retardation of rotation of a space vehicle about the center of mass for directionally fixed controlling moment,” *Cosmic Research*, Vol.13, No. 4, 1975, pp. 481-486.

[2] M. Z. Borshchevskii and I. V. Ioslovich, “Certain problems in the optimum stabilization of an axisymmetric satellite,” *Cosmic Research*, Vol. 4, No. 3, 1966, pp. 344-350.

[3] R. F. Deimel, *Mechanics of the gyroscope: the dynamics of rotation* (New York: Dover Publications, Inc., 1950).

[4] S. Dubowsky and M. D. Lichter, “Estimation of state, shape, and inertial parameters of space objects from sequences of range images,” *SPIE Conference on Intelligent Robotics and Computer Vision XXI*, Vol. 5267, Providence, RI, 2003.

[5] G. Gilardi, et. al., “Angular motion control of non-cooperative satellites using a two-arm manipulator,” *54th International Astronautical Congress of the International Astronautical Federation, the International Academy of Astronautics, and the International Institute of Space Law*, Bremen, Germany, 2003.

[6] B. B. Gragg, Jr., “The computation of approximately optimal control,” SUDAER No. 179, Department of Aeronautics and Astronautics, Stanford University, 1963.

[7] K. G. Grigor’yev and I. V. Ioslovich, “Problems of optimal control of cyclic processes,” *Soviet Journal of Computer and Systems Sciences*, Vol. 23, 1985, pp. 38-44.

[8] I. Ioslovich, “Arbitrary fuel-optimal attitude maneuvering of a non-symmetric space vehicle in a vehicle-fixed coordinate frame,” *Automatica*, Vol. 39, No. 3, 2003, pp. 557-562.

[9] I. Ioslovich, “Optimal control of rigid body rotation around center of mass,” *Journal of Dynamical and Control Systems*, Vol. 9, No. 4, 2003, pp. 549-562.

[10] I. V. Ioslovich, “Optimal stabilization of an axisymmetric satellite by means of a system of n jets (T),” *Cosmic Research*, Vol. 4, No. 4, 1966, pp. 545-551.

[11] I. V. Ioslovich, “Optimum stabilization of a satellite in an inertial coordinate system,” *Acta Astronautica*, Vol. 13, 1967, pp. 37-47.

[12] D. E. Kirk, *Optimal control theory: an introduction* (Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1970).

[13] V. F. Krotov, *Global methods in optimal control theory* (New York: Marcel Dekker, 1996).

[14] S. Matunaga, T. Kanzawa, and Y. Ohkami, “Rotational motion-damper for the capture of an uncontrolled floating satellite,” *Control Engineering Practice*, Vol. 9, 2001, pp. 199-205.

[15] R. G. Melton, D. S. Rubenstein, and H. L. Fisher, “Optimum detumbling of space platforms via a dynamic programming algorithm,” *AIAA Guidance, Navigation and Control Conference*, Williamsburg, VA, 1986.

[16] S. Yoshikawa and K. Yamada, “Impulsive control for angular momentum management of tumbling spacecraft,” *Acta Astronautica*, Vol. 60, 2007, pp. 810-819.

[17] M. Zampato, et. al., “Visual motion estimation for tumbling satellite capture,” *The British Machine Vision Conference 1996*, Edinburgh, England, 1996.