MGAnDSMs Maneuver Monte Carlo Utility

Donald H. Ellison *

Revision Date	Author	Description of Change
January 18, 2019	Donald Ellison	Created document, initial description of software utility

*Aerospace Engineer, NASA Goddard Space Flight Center, Flight Dynamics and Mission Design Branch Code 595

Contents

List of Acronyms

TCM trajectory correction maneuver

MGAnDSMs multiple gravity assist with n deep-space maneuvers using shooting

DSM deep-space maneuver

1 MGAnDSMs Missed Maneuver Analysis Strategy

At a high level, missed maneuver analysis for EMTG's chemical transcription (multiple gravity assist with n deep-space maneuvers using shooting (MGAnDSMs)) involves re-optimization of a mission-to-go starting from an off-nominal state at a critical deep-space maneuver (DSM). The initial condition can take the form of a full off-nominal 7-state generated from a navigation Monte Carlo analysis performed on the mission up to the point of the critical DSM], or just the epoch of the delayed DSM, where the initial state is obtained by propagating the pre-burn state of the spacecraft along the optimal trajectory for the length of the maneuver delay.

2 MGAnDSMs Missed Maneuver Software Utility

The Python utility `PyEMTG/SimpleMonteCarlo/MGAnDSMs_MissedManeuver.py` can be used to perform mission-to-go re-optimization for an existing MGAnDSMs mission that has been converged in EMTG. The program generates an EMTG options file based on an existing EMTG MissionOptions and Mission object pair. Besides those two inputs, the user must provide an identification string for the burn that they wish to perform the analysis on (e.g. 'j3p0b1'), an initial state to start the optimization from (including the state representation and frame), a burn delay time if the analysis is for a pure delay along the nominal trajectory, initial wait time bounds and a flag indicating whether or not a monoprop TCM is desired following the missed maneuver event.

- **original_mission**: EMTG mission object corresponding to the nominal trajectory
- **original_options**: EMTG mission options object corresponding to the nominal trajectory
- **burn_id_string**: String identifying the maneuver of interest. Must be of the form '`jXpYbZ`'
- **initial_state**: Seven entry list containing initial spaceraft state $[x, y, z, vx, vy, vz, m, \text{epoch}]$
- **initial_state_representation**: Integer corresponding to EMTG's state representation enums. One of $\{0 : \text{Cartesian}, 1 : \text{SphericalRADEC}, 2 : \text{SphericalAZFPA}\}$

- **initial_state_frame**: Integer corresponding to EMTG's frame enums. One of $\{0 : \text{ICRF}, 1 : \text{J2000\_BCI}, 2 : \text{J2000\_BCF}, 3 : \text{TrueOfDate\_BCI}, 4 : \text{TrueOfDate\_BCF}, 5 : \text{PrincipleAxes}, 6 : \text{Topocentric}, 7 : \text{Polar}\}$.
- **burn_delay**: Float indicating amount of time to delay maneuver. If ¿ 0.0, maneuver is delayed and the initial state is determined by propagating the nominal pre-DSM position/velocity. If ¡ 0.0, maneuver is early and the initial state is determined by propagating backwards from the pre-DSM position/velocity. If 0.0, then the initial state and burn epoch is fully determined by **initial_state**.
- **initial_maneuver_wait_time_bounds**: Two entry list containing the lower and upper bounds on the wait time allowed before performing the missed maneuver
- **include_TCM**: Boolean, if true, a monoprop TCM opportunity is inserted after the missed maneuver, before any subsequent DSMs and before the journey right-hand boundary.

3 Constructing an EMTG Options Object for a Missed Maneuver Scenario

The missed maneuver utility performs the following actions:

1. Removes all JourneyOptions objects that occur prior to the missed maneuver event
2. Sets the first journey's departure class to a FreePoint and its type to a launch/direct insertion
3. Fixes the departure mass
4. Includes the departure maneuver in the optimization cost function
5. Sets the pre-departure maneuver state and frame
6. Sets the departure launch window opening and first journey wait times
   (a) If a pure delay is applied, the wait time lower bound is set to the delay duration, and the upper is set to the second entry in **initial_maneuver_wait_time_bounds**
   (b) If the delay is negative (i.e. the maneuver is happening early), then the upper wait time bound is set to the delay duration and the lower is set to the first entry in **initial_maneuver_wait_time_bounds**
   (c) If a full 7-state is provided, then both wait time bounds are taken from **initial_maneuver_wait_time_bo**
7. Computes a new trialX initial guess based on the missed maneuver scenario, accounting for the truncated first journey

Figure 1 depicts a four-DSM MGAnDSMs phase, where labels marked with a superscript asterisk refer to objects and quantities associated with the nominal trajectory. The lower image depicts a missed maneuver event for DSM_1^F. The dashed line could represent a delay along the nominal trajectory or a more general off-nominal 7-state. For any missed maneuver analysis, the new EMTG mission will begin with a launch/direct insertion from a FreePoint, with the missed maneuver now handled as the phase departure maneuver. The `MGAnDSMs_MissedManeuver.py` utility offers the ability to insert a monopropellant trajectory correction maneuver (TCM) (maneuver DSM_{F_0} in Fig. 1) before the first nominal main DSM by setting **include_TCM** to true. The new initial guess will place the TCM at the half-way point between the departure maneuver and the first remaining nominal DSM.

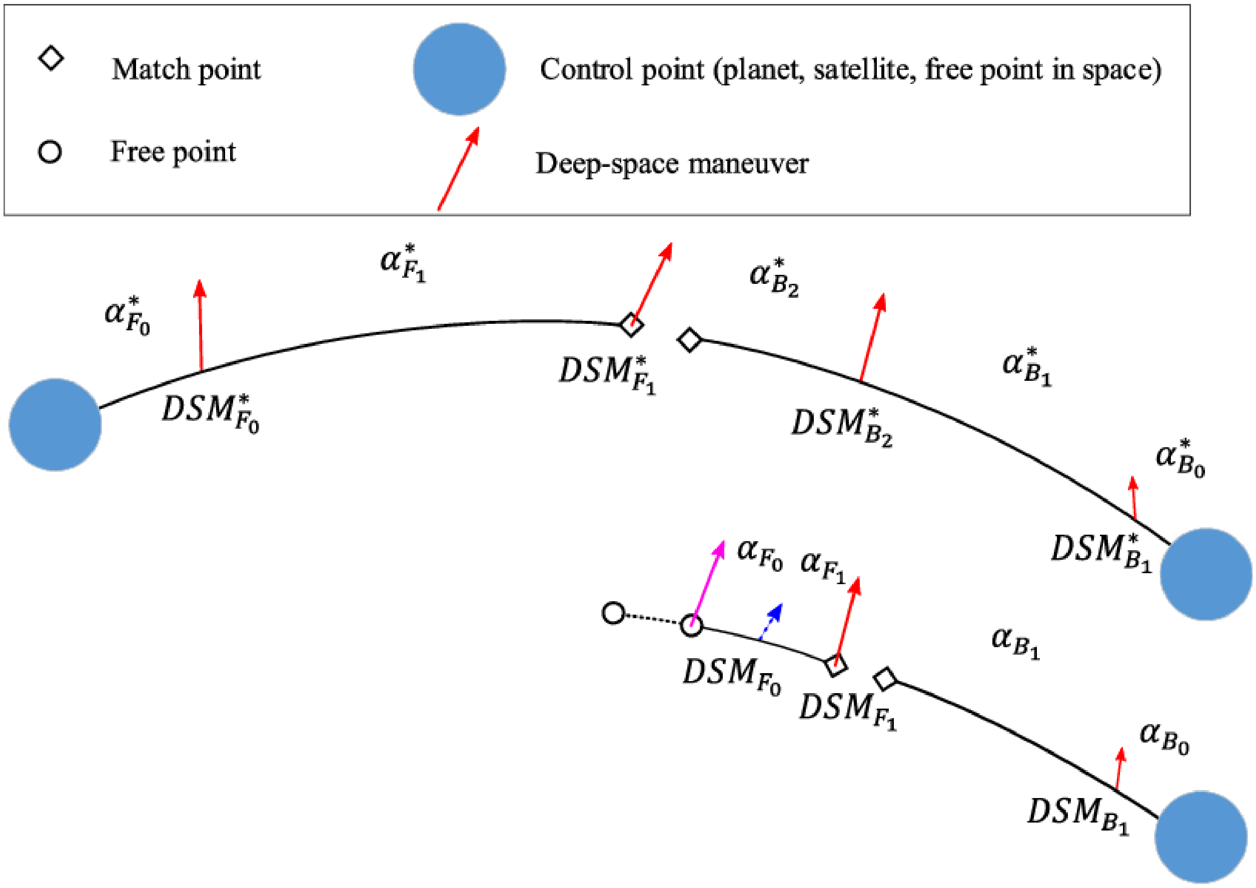

Figure 1: MGAnDSMs missed maneuver.

It is clear from Fig. 1, that the structure of the journey in which the missed maneuver occurs can change quite a bit. As a result, `MGAnDSMs_MissedManeuver.py` automatically recomputes an initial guess based on the nominal trajectory solution contained in the original MissionOptions.trialX container. Specifically, `MGAnDSMs_MissedManeuver.py` will transfer the original initial guesses for the remaining DSM components to the new guess (Fig. 2), recalculate all of the burn indices in the journey-to-go and adjust the initial guesses for the phase propellant usage variables.

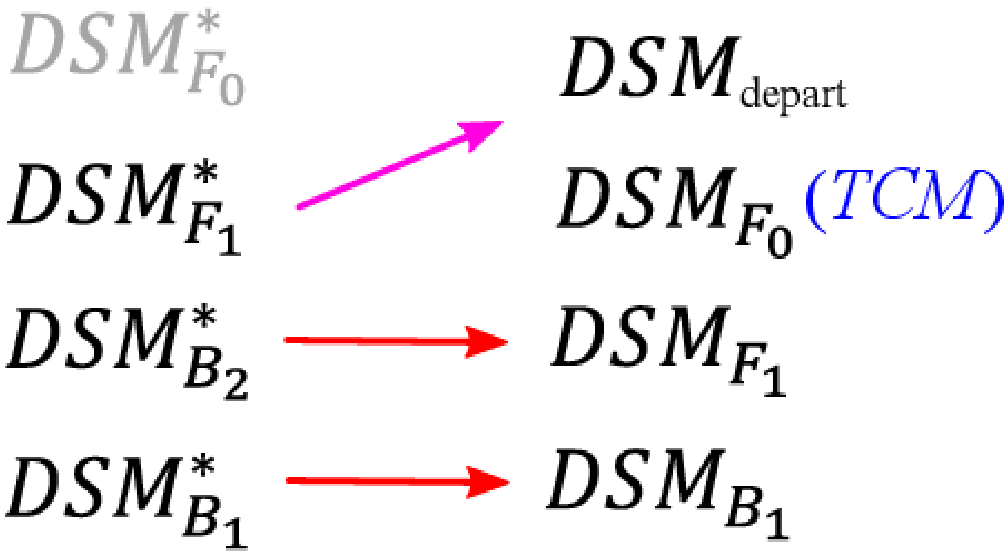

Figure 2: Transfer of maneuver initial guess information for the post-missed maneuver journey-to-go.

Computation of the new burn indices relies on two subroutines: `generateBurnIndexList` and `computeNewBurnIndices`. The former takes the integer number of DSMs allowed in an MGAnDSMs phase as an input and returns a list of burn index names for that phase in order starting at the left boundary and proceeding to the right boundary. The second subroutine returns a list of two-length lists where each two-length list contains the name of a journey-to-go burn index as the first entry and its floating point value as the second entry. The burn indices for the truncated first journey are re-computed using Eq. (1), where Δt_{p} is the flight time of the truncated journey-to-go and Δt^*_{p} is the flight time of the entire nominal journey. Should the departure maneuver have some delay applied to it, then the first burn index F_0 is computed using Eq. (2).

$$\alpha = \alpha^* \frac{\Delta t^*_{\text{p}}}{\Delta t_{\text{p}}} \tag{1}$$

$$\alpha_{F_0} = \frac{\alpha^* \Delta t^*_{\text{p}} + \Delta t_{\text{depart}}}{\Delta t_{\text{p}}} \tag{2}$$

Initial guesses for the propellant tank (propellant consumption) variables are computed using Eq. (3) and (4):

$$\Delta m_o = \sum_{k=0}^{n-1} m_k^- \left[1 - e^{\left(\frac{\Delta v_k}{g\ I_{\text{sp}}}\right)}\right] \left(\frac{1}{1 + \frac{1}{r_{\text{m}}}}\right) \tag{3}$$

$$\Delta m_f = \sum_{k=0}^{n-1} m_k^- \left[1 - e^{\left(\frac{\Delta v_k}{g\ I_{\text{sp}}}\right)}\right] \left(\frac{1}{1 + r_{\text{m}}}\right) \tag{4}$$

,

where m^- is the mass of the spacecraft immediately prior to the applied maneuver, g is the standard acceleration due to gravity, I_{sp} is the specific impulse of the chemical engine and r_m is the mixture ratio.