AAS 18-270

CLOSED LOOP GUIDANCE TRADE STUDY FOR SPACE LAUNCH SYSTEM BLOCK-1B VEHICLE

Naeem Ahmad[*], Matt Hawkins[†], Paul Von der Porten[‡], Robin Pinson[§], Greg Dukeman[¶], and Thomas Fill[‖]

The Space Launch System (SLS) Block-1B vehicle includes a low thrust-to-weight upper stage, which presents challenges to heritage ascent guidance algorithms. A trade study was conducted to evaluate two alternative guidance algorithms: 1) Powered Explicit Guidance (PEG), based on a modified implementation of PEG used on the Block-1 vehicle, and 2) Optimal Guidance (OPGUID), an algorithm developed for Marshall Space Flight Center (MSFC) and used on Constellation and other Guidance, Navigation, and Controls (GN&C) projects. The design criteria, approach, and results of the trade study are given, as well as other impacts and considerations for Block-1B type missions.

INTRODUCTION

NASA is currently building the Space Launch System (SLS) Block-1 launch vehicle for the Exploration Mission 1 (EM-1) test flight. The Block-1 vehicle consists of a Core Stage (CS) powered by four RS-25 engines, two Solid Rocket Boosters (SRBs), an upper stage known as the Interim Cryogenic Propulsion Stage (ICPS), and an Orion payload. The CS will place the ICPS and its Orion payload into a 40.7 km by 1805.7 km (22 Nmi by 975 Nmi) elliptical Earth orbit.[1] NASA's Marshall Space Flight Center (MSFC) is responsible for the Guidance, Navigation, and Controls (GN&C) algorithms, including Closed Loop Guidance (CLG), for the Block-1 CS. Since the CS has an exo-atmospheric flight profile after SRB separation similar to the Space Shuttle, Block-1 guidance utilizes the Shuttle-heritage Powered Explicit Guidance (PEG) algorithm. The Block-1 implementation of PEG has been thoroughly tested, and is robust to certain failure scenarios, including loss of a single core engine.

Meanwhile, the design of the next evolution of SLS, Block-1B, is well underway. The Block-1B vehicle is has greater payload and performance capability than Block-1, replacing the ICPS with the

[*]Aerospace Engineer, EV42/Guidance, Navigation, and Mission Analysis Branch, NASA Marshall Space Flight Center, Huntsville, AL 35812.

[†]PhD, Aerospace Engineer, EV42/Guidance, Navigation, and Mission Analysis Branch, ESSCA/Jacobs Engineering, Huntsville, AL 35806.

[‡]Aerospace Engineer, EV42/Guidance, Navigation, and Mission Analysis Branch, NASA Marshall Space Flight Center, Huntsville, AL 35812

[§]PhD, Aerospace Engineer, EV42/Guidance, Navigation, and Mission Analysis Branch, NASA Marshall Space Flight Center, Huntsville, AL 35812.

[¶]PhD, Aerospace Engineer, EV42/Guidance, Navigation, and Mission Analysis Branch, NASA Marshall Space Flight Center, Huntsville, AL 35812.

[‖]Principal Member Technical Staff, Guidance and Control Group, Charles Stark Draper Laboratory, 555 Technology Square, MS 70, Cambridge, MA 02139.

Exploration Upper Stage (EUS) powered by four RL-10 engines.[2] Planned missions for the Block-1B require EUS to perform ascent completion burns to place the EUS/payload stack into a circular Low Earth Orbit (LEO) and Translunar Injection (TLI) burns. However, the EUS has a relatively low thrust–to–weight ratio, particularly when compared with the acceleration of the CS. The low thrust of the EUS, combined with insertion at a higher altitude, creates a significantly longer ascent profile for Block-1B (on the order of 1000 seconds). This flight profile presents a challenge for any guidance algorithm, especially when considering failure scenarios such as a single engine failure during flight.

A trade study was conducted to evaluate potential alternative guidance algorithms for the Block-1B and future vehicle evolutions. Two algorithms were examined, PEG and Optimal Guidance (OPGUID). The Block-1 implementation of PEG was used as a starting point, with additional developmet to hand the Block-1B vehicle. OPGUID was a CLG algorithm developed in-house at NASA's MSFC, and was used for the Constellation program, official Block-1 ICPS in-space insight trajectories, and advanced GN&C projects.

Each algorithm was scored based on pre-selected criteria, including performance, algorithmic complexity and robustness, extensibility for potential future missions, and flight heritage. Independent reviewers determined the weightings and final scores for the various criteria.

This paper covers the design criteria, approach, and results of this trade study. In addition to the details of the trade study, this paper also shows impacts and considerations when adapting launch vehicle guidance algorithms to a wider variety of missions.

MOTIVATION/PRE-TRADE INVESTIGATIONS

The initial Block-1B guidance design began in 2014. It quickly became apparent that the expected mission class of the Block-1B vehicle was outside of the ascent mission class of both the Shuttle and SLS Block-1. From an exo-atmospheric standpoint, the Block-1 vehicle is essentially comprised of the CS whose exo-atmospheric acceleration profile (approximately in the range of one to three gs,) and duration (approximately six to eight minutes) is similar to the Shuttle. Both vehicles insert at roughly the same altitude, and both have an exo-atmospheric central burn arc that subtends around 15 degrees. The small burn arc supports a fundamental assumption in the PEG algorithm's control law that the burn is in a limited region of space where the gravitational field is nearly constant in both magnitude and direction (i.e. a flat Earth assumption). A CS engine failure produces roughly the same loss in acceleration as experienced by the loss of one of the three main Shuttle engines. So extending the PEG algorithm, with all of its successful Shuttle flight heritage, to its use on the SLS Block-1 was a natural fit. On the other hand, the Block-1B vehicle adds the EUS whose acceleration profile is dramatically lower than the CS – about 1/10 that of the CS at staging. The EUS burn more than doubles the ascent duration, increasing it by nearly 10 minutes to a duration of approximately 17-20 minutes, and increases the ascent burn arc by about 35 degrees to a total of about 50 degrees. Such burn arcs add a substantial turn rate to the thrust vector to compensate for Earth's curvature superimposed on the turning required to achieve the insertion altitude. These deviations are distinctly different from the PEG development environment of the Shuttle program. Adding off-nominal behavior and engine failures stresses the algorithm even further. Staging velocity under-speeds, resulting from early core engine shutdowns, dramatically increase the Δv requirements on the EUS, and dramatically increase the ascent duration and ascent central angle. This is because the EUS requires more than 10 times as long, compared to the CS, to achieve the same Δv. Engine failures, especially an EUS engine failure, also dramatically increase the duration

and ascent central angle, pushing the PEG algorithm design even further from some of the fundamental assumptions made during its development. The physics of the problem are challenging due to a very low thrust vehicle performing a large Δv maneuver. Early trajectory optimization analyses also had trouble finding solutions and converging, implying that the vehicle and its missions were on the cusp of feasibility.

During the initial Block-1B guidance design, the Block-1 PEG algorithm had difficulties with convergence and determining the optimal trajectories, particularly for off-nominal scenarios. Initial concerns and investigation focused on the feasibility of the vehicle to perform its mission. At this point OPGUID was brought in as an alternative. During early Block-1 analysis, long in-space trajectories were run using OPGUID for initial concept analyses. OPGUID's roots in optimal control theory, with minimal simplifying assumptions, made it a promising candidate for the type of burns and missions under investigation. It was successfully used in the first two analysis cycles for Block-1B.

At this point it was clear that there were two viable guidance options for the Block-1B vehicle. The first was to modify PEG to accommodate this new class of vehicle, by researching the derivation, resolving assumptions that were no longer applicable, and increasing fidelity where necessary. The second was to dive into the OPGUID algorithm and mature the OPGUID code for mainline use and begin the path to Flight Software (FSW) readiness. Each option had advantages and challenges. In general, the trajectory stressed each algorithm in different ways. Two small teams were formed to investigate the algorithms with the end goal of performing a trade study to determine the final algorithm to continue on for Block-1B flight and future vehicle evolutions. The remaining algorithm would be used to validate the final choice as the SLS program continues. Teams also investigated flight techniques to make the missions easier on the guidance algorithms. The results from the trade study incorporate these flight techniques into both algorithms.

ALGORITHM BACKGROUNDS

PEG Algorithm

The PEG algorithm was a creation of the Space Shuttle program for the ascent exo-atmospheric powered flight phase based on the principles of optimization theory. Roland Jaggers is generally credited with the underlying fundamental principles, derived from the calculus of variations, that form the basis of the PEG algorithm. Jaggers credited the concept to his 1960s work on the Apollo Program's Iterative Guidance Mode (IGM) and guidance theory development at NASA's MSFC.[3] In Jaggers's time this new algorithm was called Linear Tangent Guidance (LTG). Following further development by Shuttle engineers, it was renamed Powered Explicit Guidance (PEG) when the concept was selected as the algorithm for use on the Shuttle. The name is credited to Tim Brand of Draper Laboratory. It enjoyed much maturation over the years of development during the Shuttle era, forming a unified algorithm that provided guidance for not only ascent, but ascent aborts, orbit insertion, on-orbit operations, and deorbit powered burns.

PEG is explicit in the sense that it does not require a reference trajectory. The form of the algorithm's control law is based on the optimal solution to the minimum-time orbit injection problem assuming a flat Earth and a uniform gravity field on a maximum thrust arc. The maximum thrust arc implies an a-priori acceleration profile. However, this thrust profile need not be constant. The solution is a bilinear tangent steering law, which reduces to a linear tangent law when the downrange component of the position at cutoff is unconstrained. In formal vector notation, the optimal

thrust direction for the linear tangent law — the primer vector — is defined by two basis vectors representing the Lagrange multipliers in the calculus of variations solution — $\boldsymbol{\lambda}$ and $\dot{\boldsymbol{\lambda}}$ — and a reference expansion time T_λ. The optimal thrust direction at a time T is given by the primer vector $\mathbf{P}$.

$$\boldsymbol{P} = \boldsymbol{\lambda} + (T - T_\lambda)\,\dot{\boldsymbol{\lambda}}$$

The PEG solution for this law is a semi-analytical predictor/corrector algorithm. Using $\boldsymbol{v}_{go}$, the velocity to be gained by thrust, as the independent variable, and making certain approximations about the two basis vectors, allows the algorithm to analytically solve for the basis vectors and reference expansion time for the current estimate of $\boldsymbol{v}_{go}$. The resulting steering solution, combined with the burn time required to satisfy the magnitude of $\boldsymbol{v}_{go}$, is used to integrate the state forward in time to predict the burn cutoff state. At the burn cutoff position, the desired cutoff position and velocity vectors are constructed that satisfy the terminal ascent constraints. The negation of the velocity miss at cutoff (between the predicted and desired velocity vectors) is then used to correct $\boldsymbol{v}_{go}$ to drive the velocity miss to zero. The analytic solution, prediction, and correction process is repeated until the velocity miss diminishes to an acceptable level. Details of the solution can be found in References 4, 5, and 6.

This solution architecture has demonstrated flexibility, efficiency and capability over 30 years of the Space Shuttle program and was a natural candidate for continued use on the next generation of NASA launch vehicles if it could demonstrate the ability to adapt to the new vehicle configurations and trajectory profiles for nominal and off-nominal scenarios. During the trade study process, a number of adaptations were found and ultimately incorporated into the PEG code base.[7]

OPGUID Algorithm

The initial version of OPGUID was developed as a potential replacement for the IGM[8] closed-loop guidance algorithm, which at the time was the baseline guidance for the Saturn launch vehicles.[9–11] There was some concern that the simplifying assumptions in IGM might break down or result in sub-optimal propellant usage or otherwise limit the algorithm. OPGUID was also proposed for use as the Space Shuttle guidance algorithm. With the baselining of IGM for Saturn and PEG for the Space Shuttle, the OPGUID algorithm remained on the shelf until the mid-1990s when NASA's MSFC revisited the algorithm.

The computer code was translated from FORTRAN to C, along with several other changes. The unique characteristic of OPGUID is that it strictly satisfies all of the necessary conditions of optimality, including the Euler-Lagrange equations. In its original form, OPGUID numerically integrated the Euler-Lagrange equations using a 4th-order variable-step size scheme. The updated algorithm replaced the numerical integration scheme by dividing the burn into multiple equal-spaced segments and applying semi-analytical expressions to propagate the state and costate (Lagrange multipliers) over each segment.[12,13] The propagation of position and velocity over the burn requires evaluation of thrust and gravity integrals. As suggested in Reference 14, the thrust integrals have a closed-form solution assuming that the primer vector is a linear function of time. The success of IGM and PEG, which leverage that assumption over the entire burn, have demonstrated that primer linearity is a very good approximation. The gravity integrals are calculated using an average gravity approach, similar to IGM, but instead of applying a single average gravity vector over the entire burn arc, distinct average gravity vectors are applied over each of the segments.

Under the assumption of a linear central gravity field, the costate differential equations have the form of an undamped harmonic oscillator which yields costate time histories as simple functions of

initial costates, and sines and cosines of the initial position magnitude.[15] The solution to the Euler-Lagrange equations can be obtained to high precision by increasing the number of segments in which the burn arc is divided. This has the effect of providing the exact approximation-free solution to the original optimal guidance problem. A rough rule of thumb is to set the number of segments so that each one is no more than 100 seconds long, although solutions are typically not very sensitive to the number of segments. A modified multi-dimensional Newton-Raphson approach, with heuristic logic to limit the step sizes and aid convergence, is used to solve for the initial costate and burn time. The initial primer vector guess is set to a unit vector along the initial velocity vector, while the burn time guess is set using the rocket equation. The convergence behavior is relatively insensitive to the initial guess due to the step-limiting logic in the Newton-Raphson method. The Jacobian used in the Newton-Raphson method is obtained using a variational approach, which adds extra computer code but eliminates several state/costate propagations that would otherwise be necessary.[9]

TRADE PROCESS OVERVIEW

The trade process started in March 2016 with brainstorming sessions in which the SLS Guidance team created 8 major categories. Each category had various sub-categories totaling 78 Figures of Merit (FOMs). At this point, permission to proceed was requested, which was granted in October 2016. During this time, further consolidations resulted in a reduction of 8 categories to 7 categories. Due to duplication, mergers, and rejections, the total number of FOMs was reduced to 18. Each FOM would receive a score, weighted by category. Category weightings were established by in-house subject matter experts. Scores were applied at the sub-category level. Different sub-categories had different scores. Input for scoring each category was based on input from the trade study teams. Table 1 lists the categories and the corresponding sub-categories.

Table 1: Trade Study Categories and Weightings

Category	**Sub-Category(ies)**	**Weight**
Failure	Acceptable Monte Carlo Failure Rates	Pass/Fail
Programmatic Risks	Literature Availability Flight Heritage Known Flight Software Schedule Impact	8%
Assumptions and Limitations	Number and Type of Built-in Assumptions Known Limitations	3%
Extensibility/Flexibility	Ease of Developing Misison-Specific Parameters Targeting Flexibility Modularity	20%
Code Efficiency	Software Lines of Code Computational Time	16%
Algorithm Inputs	Number of Inputs	4%
Robustness	Robustness to Dispersions Convergence Reliability Iterations to Converge	21%
Objective Performance	Performance Insertion Accuracy Minimize Environmental Variations	28%

Trade Study Goals

In order to perform the Block-1B missions, the outcome of the trade study needed to result in an algorithm capable of handling a wide variety of burns, some on the order of 1000 seconds. The variety of burns encompass ascent to LEO, Apogee Raise Burns (ARBs), TLI burns, hyperbolic Earth Departure Burns (EDBs), and contingency disposal burns using low-thrust Reaction Control System (RCS) motors. With the difficulties experienced by the trade study teams in applying the Block-1 PEG algorithm to more challenging Block-1B burns, the teams desired to think ahead when choosing the winning closed loop guidance algorithm. To that end, the chosen algorithm needed to include targeting flexibility, and allow for extensibility for new potential missions. In addition, the algorithm needed to balance computational complexity, robustness, runtime, throughput and the number of built-in assumptions. The teams needed to be able to fully understand the assumptions and corresponding limitations. The winning algorithm needed to work well with multiple stages that have drastically different acceleration levels. Finally, the chosen algorithm needed to be able to respond to a single engine failure scenario.

Test Setup

For the trade study, a reference SLS Block-1B vehicle configuration was chosen. A nominal, no-failure mission was analyzed, in addition to cases with a single engine failure at selected times during flight.

The mission analyzed required a 2.5 stage ascent phase consisting of Boost Stage (BS) flight, CS flight, and an EUS Ascent burn. BS includes the thrust of the SRBs and CS engines and occurs from launch through SRB separation. CS flight goes from SRB Separation through a guidance targeted Main Engine Cutoff (MECO) to a predetermined intermediate orbit. The guidance targeted MECO is a baselined flight technique chosen to ease the difficulties associated with the low thrust-to-weight EUS. During CS flight, panels for Orion's service module and the Launch Abort System (LAS) are jettisoned after specific flight parameter constraints are reached. Finally, the EUS Ascent burn completes the ascent phase of the mission inserting EUS and its payload into a circular LEO. For the purposes of the trade study, the nominal mission is then completed after performing a 10-minute TLI burn after coasting in orbit for nearly 3 hours. This mission was considered more taxing on the guidance algorithms than the hyperbolic EDB and similar to the SRBs.

The single engine failure (i.e. engine out) scenarios modeled stressing failure times in order to test the robustness of each algorithm. The three simulated engine out scenarios were 1) an engine out just after LAS Jettison, 2) an engine out at the start of the EUS Ascent burn, and an engine out at the start of the TLI burn.

Monte Carlo (MC) analysis was used to characterize how each algorithm behaved for the nominal mission and in the failure cases. The teams were allowed two rounds to arrive at their version of the algorithm. After a first round of MC runs, the teams examined results and applied lessons learned to develop the final algorithm.

Categories for Scoring

Failure The fundamental performance standard for a guidance algorithm is the ability to complete the mission with acceptable failure rates for dispersed MC runs. Unlike other categories, this category is assigned a simple pass/fail score with no relative weighting applied, as a guidance

algorithm that cannot meet this requirement is not a viable candidate. Other scoring categories described below allow for assessing the quality of the solution, for example achieved mass-to-orbit, or accuracy in achieving insertion targets.

Given the team's experience using both algorithms it was anticipated that both algorithms would pass for the nominal mission. The SLS vehicle is required to handle loss of either a single CS engine or a single EUS engine. These scenarios present a much more difficult problem and are more challenging for the guidance algorithm. The feasibility of reaching primary or alternate targets for any arbitrary engine out scenario was not known in advance, so the algorithms were compared for engine loss across different regions of flight. If a region was found where only one algorithm could achieve insertion, this result would be used to select the algorithm.

Programmatic Risks Programmatic risks are non-technical risks that are generally within the control of the program or project office. These risks can be associated with program estimation such as cost estimation and schedule estimation. Furthermore, programmatic risks are also associated with program planning, execution, and communication.*

To quantify these programmatic risks, each algorithm's literature availability, flight heritage and known FSW schedule impacts were assessed. The literature availability sub-category, focuses on reference materials in order to ensure sufficient literature is available. In the flight heritage sub-category, the team assessed the extent to which each algorithm is new or is inherited from other programs and looked at Technology Readiness Levels (TRLs). TRL, as indicated by Figure 1, categorizes the maturity level in ranges of 0-9 where 0 indicates that the technology only exhibits a basic principle and 9 indicates the highest maturity level in which the technology has been adopted as part of a mission operations process. In addition to algorithm maturity, flight heritage data was available for each algorithm.The FSW impact sub-category focuses on the percentage of code that needs to be brought under FSW coding standards, required documentation delivery, and any requirement changes that could impact schedule and cost.

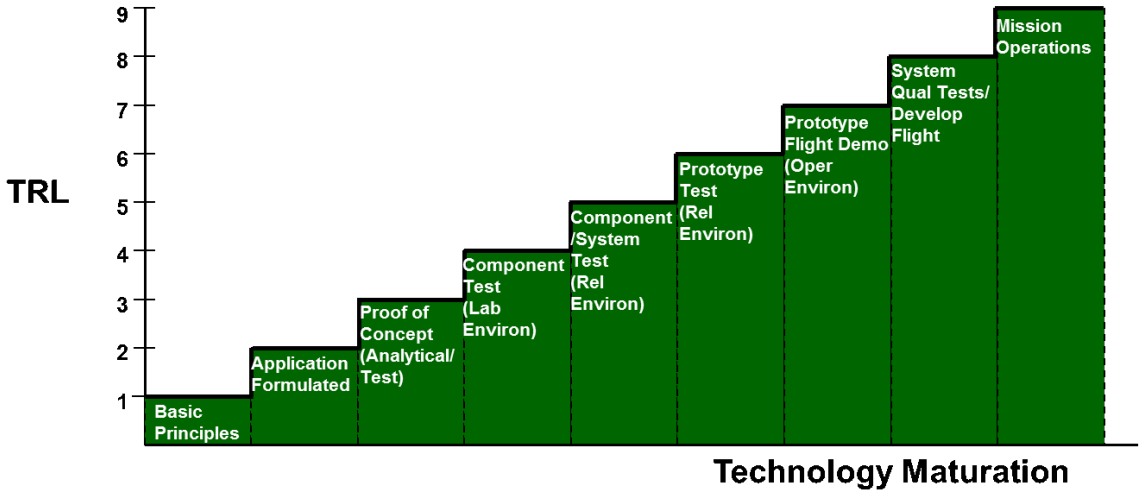


Figure 1: Technology Readiness Level

*https://www.dau.mil/glossary/pages/2589.aspx

Assumptions and Limitations The selected algorithm needs not only to support the Block-1B configuration, but also needs to support future missions that have not yet been envisioned. With this regard, it his highly important that each algorithm's assumptions (e.g. uniform gravity) and limitations (e.g. constant acceleration phases) are well understood. Working with subject matter experts, the team identified a number of assumptions made in both of these algorithms, as well as each algorithm's limitations.

Extensibility and Flexibility The selected algorithm needs to be flexible enough to support SLS missions beyond Block-1B. In order to assess this, each algorithm's modularity, targeting flexibility, and ease of mission development were evaluated.

Modularity focuses on how easy it is to replace components within each algorithm. This includes features such as the ability to interchange gravity models, modify predictor routines, select a different target set, and specify different forms of constraints. Targeting flexibility focuses on how to add a new set of targets in each algorithm to support different missions — for example, adding a de-orbit burn or an EDB. Lastly, ease of mission development focuses on the number of inputs associated with each algorithm that can change not only mission-to-mission, but even from vehicle configuration to vehicle configuration within a specific mission (e.g. light or heavy payloads)

Code Efficiency Flight computers have improved significantly since the early Apollo days. Due to the time, effort, and cost that it takes to qualify a computer to withstand space environments, there are a limited number of computers that can satisfy flight requirements and their processing capability, speed, and throughput significantly lags the standard computing power currently used every day. The algorithm's code will have to complete all its calculations in a given time allotted and work efficiently.

Two measurements were used to judge this category. The first was Software Lines of Code (SLOC). This was conservatively estimated for each algorithm and included comment lines as these would need to be maintained along with the code. It did not include header files, structure definitions, and utilities as these would be the similar for the two algorithms. The second measure was an estimate of computational time. For this category, it was assumed the algorithm would have a 1 Hz process in which to operate. If the algorithm used a small portion of the 1 Hz operational time it received the full score.

Algorithm Inputs All launch vehicle CLG algorithms require vehicle data and tuning parameters as inputs. However, the greater the number of these inputs and tuning parameters, the greater will be the complexity and maintainability concerns. Therefore, having fewer algorithm inputs is preferable.

The metric for rating this category was based on a simple count of the number of input parameters. The algorithm that required fewer inputs received the full points for this category, while the other received zero. If the number of inputs for each algorithm was within 2 of each other, a tie was declared and each algorithm scored equally.

Robustness This category has three sub-categories: robustness to dispersions, convergence reliability, and iterations to convergence. A key FOM of guidance algorithms is robustness in the presence of dispersions. Dispersions occur due to model and event timing uncertainties or due to unexpected anomalies such as an engine-out failure. On SLS, the FSW allows 10 iterations within each 1 Hz guidance cycle before non-convergence is declared and an attitude hold or other backup steering mode is implemented for a full cycle. Although convergence can still occur on subsequent cycles and preserve mission success, it is strongly desired that the algorithm always converge within

its allotted number of iterations. It is desired that the algorithm converges reliably even in the presence of anomalies, particularly when part or all of the mission objectives can still be preserved. Although fewer iterations are preferred, this FOM is a less meaningful discriminator when both algorithms reliably converge within the allocated number of iterations per guidance cycle.

Objective Performance The key purpose of a launch vehicle CLG algorithm is to provide steering law parameters that minimize propellant usage (i.e. maximize mass-to-orbit performance, or simply "performance") through a burn, while accurately inserting the vehicle into the desired orbit. A secondary objective is to minimize environmental variation (e.g. the spread of potential vehicle body rates at engine cutoff), though flight techniques and engine performance are much bigger players than the choice of the closed loop algorithm. This category contained quantitative and qualitative evaluations of these primary and secondary objectives: performance, insertion accuracy, and minimization of environmental variations.

Performance was measured by evaluating the total mass-to-orbit at orbit insertion using a simulation assuming no failures occur. The algorithm that achieved the better performance received three points, while the lesser performing algorithm received zero, one, or two points depending on the disparity. If the algorithms achieved within 91 kg (200 lbm) of each other, they were deemed equivalent and both received equal points.

Insertion accuracy was assessed by comparing how accurately the vehicle achieves the ascent and TLI orbits. Specifically, apogee error, semi-major axis error, and wedge angle* were assessed for ascent, while semi-major axis error, argument of perigee error, eccentricity error, and wedge angle were assessed for TLI. Up to five points were awarded to each algorithm depending on how well the vehicle achieved each orbit parameter against the intended target.

To determine how the algorithms minimized environmental variations, each algorithm was evaluated qualitatively by analyzing time history trajectory plots of parameters such as angle-of-attack at core stage engine cutoff, body rates at CS and EUS stage engine cutoff, etc. Up to one point was awarded to each algorithm. Should one algorithm have reduced trajectory variations much more than the other algorithm, the losing algorithm was awarded zero points.

RESULTS

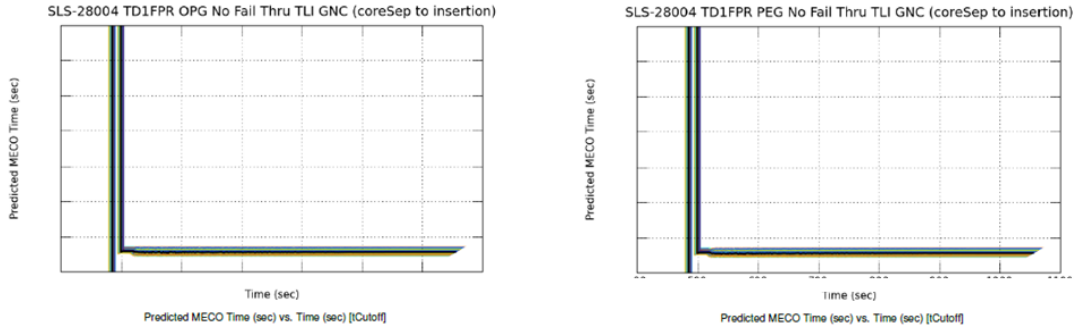
Failure Both algorithms performed all trade study MCs with acceptable failure rates. Early core engine out scenarios proved challenging to both algorithms, and no region of flight was found where only one algorithm was able to perform.

One noticeable difference in behavior was the prediction of cutoff time for engine out scenarios. During a particularly stressing engine out case, OPGUID was able to successfully predict a new cutoff time, however PEG was required to use a Shuttle-heritage elevation limit. In this case PEG's predicted cutoff time slowly increased over several hundred seconds of EUS flight, before finally leveling off. The final cutoff time was determined more than 200 seconds in advance, indicating that PEG was not in danger of running up against an artificial limit at Second Engine Cutoff 1 (SECO-1). Because both the cause and effect are well understood, this issue did not warrant violating the pass/fail criteria.

Figure 2 shows the predicted SECO-1 times — cutoff time of the EUS engines for the ascent burns — for a nominal MC run. Both algorithms predict a nearly constant cutoff time. In compar-

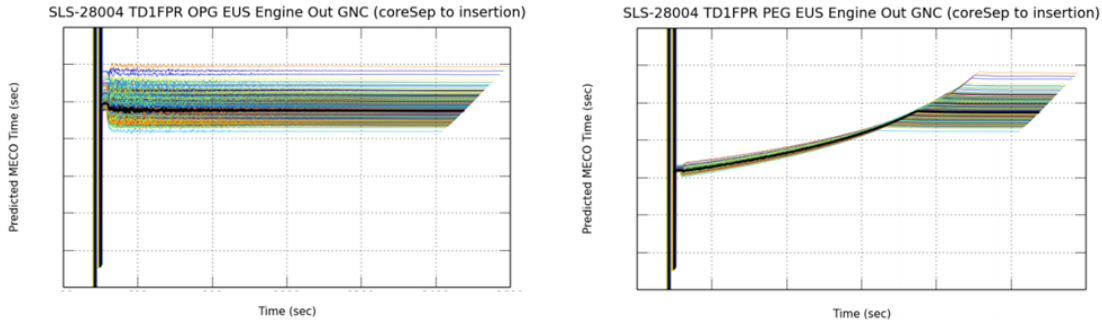
*The wedge angle is the angle between the targeted orbit plane and the actual orbit plane.

ison, Figure 3 shows the predicted SECO-1 times for a MC run with loss of an EUS engine at the beginning of EUS flight (e.g. the engine fails to ignite). Note that OPGUID's prediction is constant throughout, while PEG's prediction grows before finally settling to a constant output.



(a) OPGUID (b) PEG

Figure 2: Predicted SECO-1 times for a nominal mission.



(a) OPGUID (b) PEG

Figure 3: Predicted SECO-1 times for a single engine failure at the start of the EUS Ascent Burn.

Programmatic Risks PEG won the programmatic risks category. 47% of PEG's code contained heritage from Shuttle, whereas OPGUID did not feature any flight heritage. PEG's Shuttle heritage and its implementation on the Block-1 design resulted in less code needing to be brought under FSW standards compared to OPGUID. Both of these algorithms had the same TRL score of 6 and availability of a sufficient amount of literature.

PEG, with its Block-1 heritage, is ahead of OPGUID in that it has been refined per the FSW coding standard. There are a few changes implemented in PEG in order to handle specific limitations specified earlier in this section. OPGUID has not been implemented from the FSW perspective, which means it would require significant effort to achieve a FSW-ready level, leading to greater impact on the schedule. It has extraneous code that would need to be culled before it is FSW-ready.

Assumptions and Limitations OPGUID won in the assumptions and limitations category. Since OPGUID is derived using Euler-Lagrange equations and the principle of optimal control theory, it makes fewer assumptions leading to fewer limitations. Table 2 and Table 3 list the assumptions and limitations for both algorithms.

Table 2: Algorithm Assumptions

PEG	OPGUID
1. Each subphase is either constant thrust or constant acceleration.	1. Each subphase is either constant thrust or constant acceleration.
2. Designed to work in the exo-atmospheric region of flight.	2. Designed to work in the exo-atmospheric region of flight.
3. The PEG steering law formulation assumes a flat Earth (analogous to the uniform gravity field). This generally restricts use of the algorithm to limited thrust arcs in ascent applications with an altitude constraint.	
4. The primer vector and its time-rate of change are assumed to be orthogonal.	
5. PEG's approach to the corrector is effectively a Newton-Raphson method that assumes the Jacobian matrix is approximately the identity matrix. This approximation breaks down for long burn arcs, requiring a half-step correction to prevent oscillations.	
6. PEG's thrust modeling in the predictor assumes the acceleration due to thrust is the form of a quadratic polynomial.	

Table 3: Algorithm Limitations

PEG	OPGUID
1. Each subphase ends based on either time or mass.	1. Each subphase ends based on either time or mass.
2. Large thrust arcs must be limited because of the flat Earth assumption.	
3. For the class of EUS burn arcs, when the elevation limit constraints are on, PEG under-predicts the cutoff time by quite a bit for large burn arcs.	
4. The accuracy of using a neighboring coasting trajectory for predicting the effects of gravity on position and velocity over the thrust arc degrades for large thrust arcs.	
5. The PEG correction process is predicated on the desired inertial velocity at cutoff being much less sensitive to changes in the velocity-to-be-gained (by thrust) vector than the predicted cutoff velocity. If that is not the case, PEG convergence performance will degrade, requiring the use of more complex correction methods.	

Extensibility/Fexibility In the extensibility/flexibility category, both algorithms drew a tie. During the CLG trade pre-work, including thorough examination of the algorithms, OPGUID and PEG were shown to be modular. They were fairly evenly matched in terms of modularity with the major routines and sections broken up so new functions or switch/case statements can be easily inserted. Both algorithms have the ability for the predictor to change from an analytical or semi-analytical to a full numerical predictor with a choice of propagation routines. Gravity models can be swapped out, if ascent is from the moon or Mars, or if a higher fidelity Earth model is required. The corrector routine itself is modular; however, certain parts would more likely be interchanged. OPGUID is flexible to the number of states to be corrected (which it does when constraints are relaxed) and the linear algebra routine which calculates the correction. PEG can switch out the Newton-Raphson correction step calculation to an alternate method, if desired. Moreover, both algorithms have the ability to relax certain portions of the targeted constraints when needed in flight, especially near the end of flight to prevent over-constraining the problem. Additionally, a targeting flexibility test was performed where a new targeting mode was created in both PEG and OPGUID. From the targeting flexibility analysis, each team demonstrated a straightforward approach to develop new targeting modes for each algorithm. This provided confidence that each algorithm can be extended to potential missions requiring new targeting schemes. Finally, both algorithms had a similar amount of mission-specific parameters that need to be adjusted throughout variations in vehicle and missions.

Code Efficiency PEG was the clear winner in the code efficiency category. The simplifying assumptions that enabled it to work on a 1970s-era computer, along with years of program modifications and applying FSW coding standards for the Block-1 vehicle, lead to its advantage over OPGUID. Furthermore, the count of SLOC for PEG was 60% of the OPGUID count, requiring less work for FSW readiness and maintenance. Finally, OPGUID took an order of magnitude longer than PEG to complete its calculations. However, both OPGUID and PEG took a small fraction of the 1 Hz cycle to complete their calculations. Therefore, both algorithms received the point for operating well inside a 1 Hz frame.

Algorithm Inputs The algorithms received a tied score in the algorithm inputs category. The final tally of algorithm inputs came out as 17 for PEG and 15 for OPGUID. Since the teams determined that having counts within two of each other is equivalent, both algorithms were awarded full points for this category.

Robustness Both algorithms demonstrated reliable convergence within a reasonable number of iterations in every simulation. PEG generally requires fewer iterations to initially converge, as well as at dynamic events such as when an engine failure occurs. The same is true during steady state flight, where PEG typically requires one or two iterations vs. OPGUID's three or four iterations. OPGUID's convergence tolerances and corrector step-limiting parameters are set in a fairly conservative manner, so as to favor convergence reliability over computational efficiency. It is likely that OPGUID's iteration counts could be reduced, while retaining convergence reliability, by tuning such parameters. In the final scoring, OPGUID and PEG were deemed to have equivalent performance in this category.

Objective Performance The algorithms received a tied score in the objectives performance category. Performance, insertion accuracy, and minimization of environmental variations were all deemed the same or similar.

As Table 4 shows, the mass-to-orbit performance of both algorithms were nearly identical and were well within the 91 kg (200 lbm) threshold used to indicate similar performance. Both the

average and the 3-sigma low performing value using a MSFC standard 90% confidence level[16] displayed these results. Table 5 shows that for the more stressing engine out cases, OPGUID showed better performance for the post-LAS jettison engine out case, while PEG performed better for the EUS Ascent and TLI engine out cases. Failure cases, including engine outs, are assessed at a 2-sigma level rather than 3-sigma.[16] At the conclusion of the trade study, the trade study teams were not able to resolve any of these differences. Performance is a secondary concern for engine failure scenarios and neither algorithm displayed a definitive advantage over the other for these stressing cases. Therefore, the teams deemed that both algorithms displayed equivalent performance.

Likewise, the insertion accuracy was similar between the two algorithms. Table 6 lists the nearly identical in-plane and out-of-plane insertion errors for both ascent and TLI insertion using the two algorithms for the nominal mission simulation. Note that the wedge angle is a one-sided parameter and uses a different percentile than two-sided parameters.

Finally, both algorithms led to similar trajectory variations. The engine cutoff body rates were similar for both algorithms. As an example, the ascent pitch rate profiles for each algorithm, illustrated in Figures 4 and 5, showed similar behavior.

One interesting result that did display some different behavior was in the different EUS Ascent burn pitch profiles as shown in Figure 6. Note that the data in the left plot in Figure 6 was scaled to match the scaling of the right plot, to illustrate the different pitch profiles observed by the two algorithms. The focus on the plots should be the on the different shapes of the pitch profiles, instead of grid lines. This behavior is a result of the long EUS ascent burn arc and constrained LTG steering law of PEG versus the steering law that OPGUID follows by constraining the necessary conditions of optimality. The ascent injected mass differences between the two algorithms shown in Table 4 indicate that either steering law equates with the same performance, even though the steering laws prescribe slightly different paths.

Table 4: Nominal Mission Performance Results

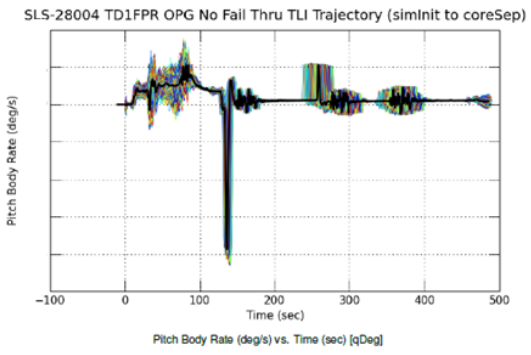
Parameter	Difference between PEG and OPGUID (PEG − OPGUID)	
	Average	99.865% w/ 90% CL Low
Ascent Injected Mass, kg	4	-7
TLI Injected Mass, kg	14	5

Table 5: Engine Out Performance Results

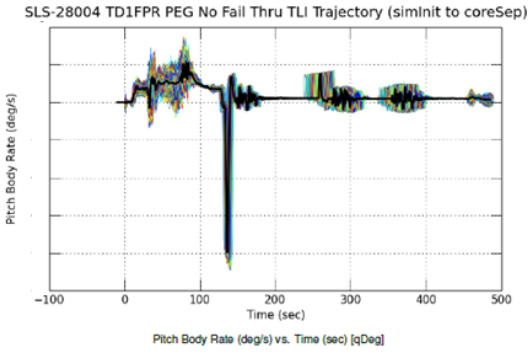
Parameter	Difference between PEG and OPGUID (PEG − OPGUID)	
	Average	97.725% w/ 90% CL Low
Post-LAS Jettison Engine Out		
Ascent Injected Mass, kg	-716	-647
TLI Injected Mass, kg	n/a	n/a
EUS Ascent Burn Start Engine Out		
Ascent Injected Mass, kg	73	367
TLI Injected Mass, kg	n/a	n/a
EUS TLI Burn Start Engine Out		
TLI Injected Mass, kg	374	369

Table 6: Nominal Mission Insertion Accuracy Results

Parameter		99.865% w/ 90% CL	
		Low	High
Ascent Insertion In-Plane Errors			
Apogee error, km	OPGUID	0.02	0.44
	PEG	-0.02	0.35
Semi-major axis error, km	OPGUID	-0.35	0.19
	PEG	-0.31	0.17
TLI Insertion In-Plane Errors			
Semi-major axis error, km	OPGUID	-565	374
	PEG	-550	359
Argument of perigee error, deg	OPGUID	-0.009	0.005
	PEG	-0.009	0.006
Eccentricity error	OPGUID	0.0002	0.0004
	PEG	0.0002	0.0003
		99.73% w/ 90% CL	
		Low	High
Ascent Insertion Out-of-Plane Error			
Wedge angle, deg	OPGUID	n/a	0.00111
	PEG	n/a	0.00019
TLI Insertion Out-of-Plane Error			
Wedge Angle, deg	OPGUID	n/a	0.0027
	PEG	n/a	0.0027

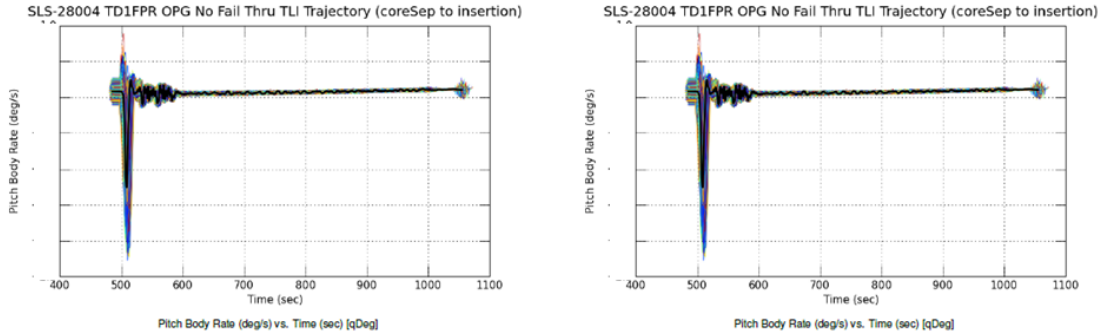


(a) OPGUID



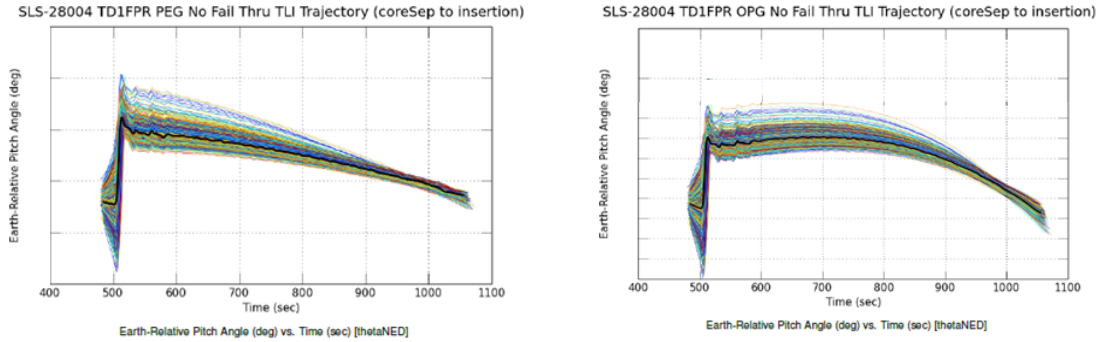
(b) PEG

Figure 4: Nominal mission Pitch Body Rate profile from Liftoff through CS Separation.



(a) OPGUID (b) PEG

Figure 5: Nominal mission Pitch Body Rate profile from CS Separation through Ascent insertion.



(a) OPGUID (b) PEG

Figure 6: Nominal mission Earth Relative Pitch Angle profile from CS Separation through Ascent Insertion.

Final Trade Study Results

Table 7 shows the results of the final scoring, with weighting and scoring by category. The scoring is designed so that the total scores sum to 100.0. Each category's weight is divided up between each algorithm depending on the margin by which each algorithm won the specific category.

Table 7: Trade Study Results

Category	**Weight**	**Score**	
		PEG	OPGUID
Failure	Pass/Fail	Pass	Pass
Programmatic Risks	8%	5.2	2.8
Assumptions and Limitations	3%	0.4	2.6
Extensibility/Flexibility	20%	10	10
Code Efficiency	16%	9.6	6.4
Algorithm Inputs	4%	2	2
Robustness	21%	10.5	10.5
Objective Performance	28%	14	14
Total		51.7	48.3

The final scores between the two algorithms demonstrated that the algorithms are fairly well matched given the weighting and importance applied to the different categories. In the three most important categories (according to reviewer weighting), PEG and OPGUID were tied. PEG's score was higher than OPGUID's on the next two most important categories, leading to PEG receiving a slightly higher score overall. Given the closeness of the scores, the final decision was made by an independent team consisting of SLS Vehicle Management Leadership. The modified Block-1B PEG was chosen as the guidance algorithm for Block-1B, in-line with the final results of the trade study.

CONCLUSION/FUTURE WORK

In order to select between potential CLG algorithms for the future Block-1B design, a trade study was conducted with predefined objectives. These objectives were derived from desired Block-1B capabilities which included: the ability to perform multiple burns such as ascent, TLI, and EDB; the ability to handle long burn arcs; the ability to support multiple stages with each stage having drastically different acceleration levels; flexibility to various types of targeting; an appropriate level of accuracy for models such as thrust and gravity; and a balance among computational complexity, robustness, runtime, and the number of known assumptions and limitations.

The trade study demonstrated that both algorithms met objective performance similarly. For example, achieved target accuracy, trajectory variations, and realized mass-to-orbit between these algorithms were similar. Both algorithms were shown to be robust for the challenging Block-1B class of burns. However, PEG is less complex and therefore will require less work for FSW readiness and maintenance. OPGUID's use of the full Euler-Lagrange equations increases its complexity compared to PEG's pseudo-analytic formulation. At the conclusion of the study, SLS Vehicle Management recommended PEG as the choice of algorithm for Block-1B.

The trade study showed that OPGUID is a viable guidance algorithm, though lacking in maturity. Since the completion of the trade study, work has continued on OPGUID at a low effort level and

will continue as time allows. The immediate goal is to mature the code using coding standards and streamline the code to decrease SLOC. Long term plans include looking for possible demonstration tests to increase the TRL and prepare it for use on other flight projects, should the opportunity arrive. In addition, it will continue to be used as an independent check on Block-1B PEG.

Finally, the trade study experience provided significant insight into the behavior of both OPGUID and PEG. As a result of algorithm development for the trade study, a number of improvements were incorporated into the PEG code base, increasing its robustness and reliability.[7]

REFERENCES

[1] B. Donahue et al., "Space Launch System: Development Status," AIAA SPACE 2016, Sept. 2016.

[2] W. H. Gerstenmaier, "Human Exploration & Operations Update on Mission Planning for NAC," NASA Human Exploration and Operations, NASA Advisory Council, Nov. 2016.

[3] J. Goodman, "Powered Flight Guidance History and Bibliography," Tech. Rep. JSC-35146, NASA Johnson Space Center, 2013.

[4] R. L. McHenry, T. J. Brand, A. D. Long, B. F. Cockrell, and I. J R Thibodeau, "Space Shuttle Ascent Guidance, Navigaiton, and Control," *The Journal of the Astronautical Sciences*, Vol. 27, Jan.- 1979, pp. 1–38.

[5] T. J. Fill, "Introduction to Bi-Linear Tangent Steering for Shuttle Ascent and Aborts," Technical Memorandum EGB-89-108, SHUTTLE-89-022, The Charles Stark Draper Laboratory, Mar. 1980.

[6] R. Jaggers, "Shuttle Powered Explicit Guidance (PEG) Algorithm," JSC Internal Note 75-FM-61 JSC-26111, NASA Johnson Space Center, Nov. 1992.

[7] P. Von der Porten, N. Ahmad, M. Hawkins, and T. Fill, "Powered Explicit Guidance Modifications and Enhancements for Space Launch System Block-1 and Block-1B Vehicles," *AAS 18-136*, 2018.

[8] D. C. Chandler and I. E. Smith, "Development of the Iterative Guidance Mode with Its Application to Various Vehicles and Missions," *Journal of Spacecraft and Rockets*, Vol. 4, 1967, pp. 898–903.

[9] K. R. Brown, E. F. Harrold, and G. W. Johnson, "Rapid Optimization of Multiple-Burn Rocket Flights," *NASA CR-1430*, Oct. 1969.

[10] K. R. Brown, E. F. Harold, and G. W. Johnson, "Some New Results on Space Shuttle Atmospheric Ascent Optimization," *AIAA Paper No. 70-978*, 1970.

[11] A. O. Cohen and K. R. Brown, "Real-Time Optimal Guidance for Orbital Maneuvering," *AIAA Journal*, Vol. 11, Sept. 1973, pp. 1266–1272.

[12] G. Dukeman, "Atmospheric Ascent Guidance for Rocket-Powered Launch Vehicles," *AIAA Paper No. 2002-4559*, Aug. 2002.

[13] G. Dukeman and A. Calise, "Enhancements to an Atmospheric Ascent Guidance Algorithm," *AIAA Paper No. 2003-5638*, Aug. 2003.

[14] R. R. Burrows and G. A. McDaniel, "A Method of Trajectory Analysis with Multi-Mission Capability and Guidance Application," *AIAA Paper No. 68-844*, Aug. 1968.

[15] D. J. Jezewski, "Optimal Analytic Multiburn Trajectories," *AIAA Journal*, Vol. 10, Oct. 1972, pp. 680–685.

[16] J. M. Hanson and B. B. Beard, "Applying Monte Carlo Simulation to Launch Vehicle Design and Requirements Analysis," Tech. Rep. TP-2010-216447, NASA Marshall Space Flight Center, Sept. 2010.