

Application of a Dual-Quaternion Six Degree-of-Freedom Guidance to Human-Scale Mars Entry, Descent, and Landing

Rafael A. Lugo¹ and Soumyo Dutta²

NASA Langley Research Center, Hampton, VA, 23681, USA

Michael S. Manwell³

Analytical Mechanics Associates, Hampton, VA, 23666, USA

Landing humans on Mars comes with many challenges, including the execution of a safe and precise entry, descent, and landing (EDL) sequence. Various NASA studies have shown that there are a variety of EDL guidance methods that potentially offer solutions to the human-scale EDL precision landing problem and work is ongoing to assess new and novel methods. As part of these ongoing studies, a dual-quaternion-based six degree-of-freedom guidance algorithm was implemented in the Program to Optimize Simulated Trajectories II (POST2), a NASA- and industry-standard spacecraft trajectory and vehicle design tool. This algorithm considers both translational and rotational dynamics and casts the trajectory optimization problem as a quadratically constrained quadratic program (QCQP) with various constraints at discrete nodes throughout the trajectory. The QCQP problem is solved via an alternating direction method of multipliers (ADMM) approach, and the output is a discretized optimal trajectory. The guidance is applied to a NASA reference human-scale Mars EDL system, and results are compared and discussed.

I. Introduction

Perhaps the most significant challenge in landing humans on Mars is the execution of a safe and precise entry, descent, and landing (EDL) sequence. The current state-of-the-art Mars EDL guidance algorithms are derived from Apollo-era approaches and are unlikely to offer the robust flexibility that high-mass human-scale EDL systems will require. Various NASA efforts such as the Entry, Descent, and Landing Architecture Study (EDLAS) and the Entry Systems Modeling (ESM) project have shown that there are a variety of EDL guidance methods that potentially offer solutions to the human-scale EDL precision landing problem and work is ongoing to assess new and novel methods [1, 2, 3, 4, 5].

As part of these ongoing studies, in 2018 NASA funded three Early Career Faculty (ECF) awards to advance these new and novel guidance algorithms [6, 7]. To that end, Dai et al. developed a dual quaternion (DQ)-based six degree-of-freedom (6DOF) guidance algorithm [8, 9] that considers both translational and rotational dynamics and casts the trajectory optimization problem as a quadratically constrained quadratic program (QCQP) with various constraints at discrete nodes throughout the trajectory. The QCQP problem is solved via an alternating direction method of multipliers (ADMM) approach, and the output is a discretized optimal trajectory. In this work, this guidance was implemented in the Program to Optimize Simulated Trajectories II (POST2) and applied to a NASA reference human-scale Mars EDL system. For the remainder of this work, the guidance algorithm will be referred to as the DQ-ADMM guidance.

¹ Aerospace Engineer, Atmospheric Flight & Entry Systems Branch, AIAA Senior Member.

² Aerospace Engineer, Atmospheric Flight & Entry Systems Branch, AIAA Associate Fellow.

³ Aerospace Engineer, Atmospheric Flight & Entry Systems Branch, AIAA Member.

II. DQ-ADMM Algorithm

Dual quaternion-based methods have seen a recent proliferation in the fields of spacecraft guidance and navigation [10, 11, 12, 13]. The method utilized here consists of three distinct steps that are summarized as follows. A more detailed derivation and discussion is available in [8] and [9].

The first step in the DQ-ADMM scheme formulates the problem into a QCQP. At guidance initiation, POST2 passes relevant information to the guidance so it can integrate the DQ equations of motion across 31 temporal nodes. This information includes current vehicle states, mass properties, and aerodynamic reference area. This step also includes transforming state variables from POST2 specific coordinate frames to coordinate frames unique to the DQ-ADMM. Planet model information is also passed to the guidance in the form of gravitational constant, equatorial radius, and planetary rotational constant. The density model used for calculating aerodynamic forces within the guidance is a user defined input in the form of a 7th order polynomial, chosen to facilitate calculation speed.

The guidance then normalizes the received inputs and performs a coarse integration that covers the nodal space from the time of initial guidance call to the estimated time at which the vehicle reaches the altitude to begin its powered descent. The solution from this coarse integration across the 31 temporal nodes is the initial guess of the normalized solution vector. This step also constructs the equality and inequality constraint matrices that are based on the DQ equations of motion and operational/mission constraints respectively (e.g., attitude, maximum structural loading, and heat flux).

The second step employs a customized ADMM solver which attempts to solve for the objective function while minimizing differences on the consensus, equality, and inequality constraint equations. The solver employs a classical ADMM scheme in which the primal and dual variables are updated along with the penalty coefficients before beginning the next iteration. The solver iterates until the error on the constraint equations converges below a user defined tolerance, or a maximum number of iterations is reached to prevent infinite loops in the case convergence cannot be achieved. Note that this constraint tolerance is input as a scalar and is applied to all constraints. References [8] and [9] contain comprehensive descriptions of the customized ADMM algorithm.

The third step, while not internal to the guidance itself, is still required when calling the guidance from an integrated simulation environment. This step derives the spacecraft guidance commands from the converged nodal solutions. The unnormalized solution vector is transformed to a frame and format usable by the POST2 control and steering systems and linearly interpolated across time to provide instantaneous commands to the outer-loop trajectory as an open-loop system.

III. Integration with POST2

As part of the ECF program, the code authors provided NASA a version of the DQ-ADMM codebase to implement in POST2. POST2 is a NASA- and industry-standard flight dynamics simulation tool [14] that has been used to design and evaluate many Earth and Mars flight and test systems [15, 16, 17].

To integrate the guidance algorithm into POST2, the codebase had to first be converted to a format compatible with the C/C++ POST2 program. Multiple approaches were considered, including the use of MATLAB's Autocode feature to automatically translate MATLAB code to C language. However, since the provided algorithm was tailored to a specific problem used by the ECF team, in this work, the DQ-ADMM MATLAB code was manually translated to C++ and verified against the original code (discussed in Section III.B). The Armadillo library [18, 19] was utilized due to its MATLAB-like syntax to facilitate implementation of the extensive linear algebra with very large matrices.

The POST2 simulation incorporates a variety of environment, vehicle, and guidance, navigation, and control (GN&C) models. In this work, Mars gravity and atmosphere were modeled, as well as the entry system and associated aerodynamics. The DQ-ADMM algorithm was called via POST2's flight software interface, described in the next section, and converged commands were passed back to the simulation 3DOF steering algorithms in the form of an open-loop system. The outer-loop simulation then continued with the vehicle attempting to follow the provided commands throughout the atmospheric entry phase. Additionally, the inner loop nodal guidance solution was saved for later comparison to the outer-loop, as-flown, trajectory. A diagram illustrating the integration of the DQ-ADMM guidance algorithm with POST2 is shown in Fig. 1.

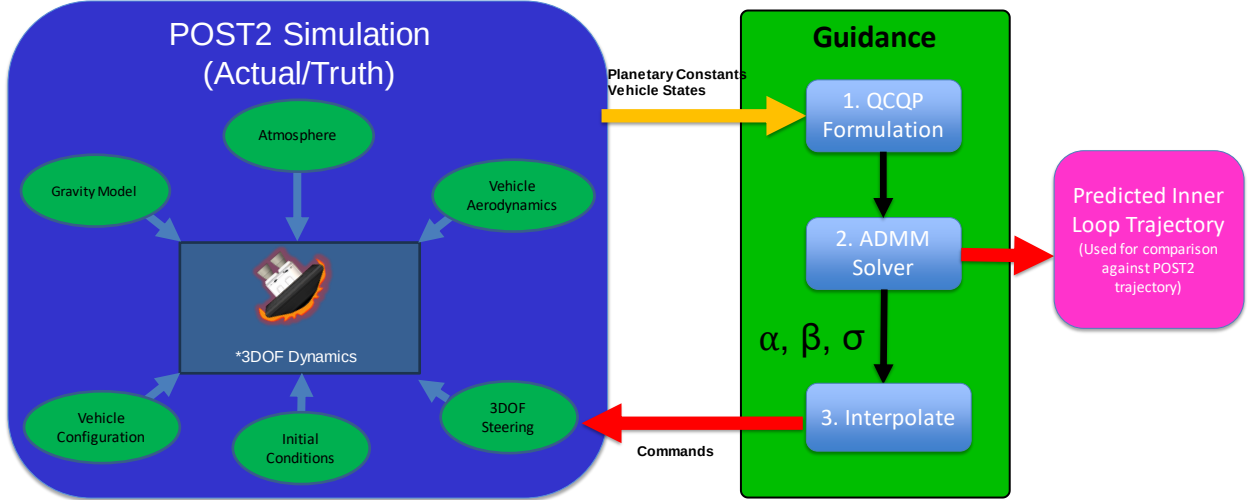


Fig. 1 Diagram of POST2 DQ-ADMM Integration.

A. POST2 Flight Software Interface

To simulate a spacecraft flight computer, the POST2 flight software interface was utilized. This interface simulated call-scheduling at user-specified rates and order. For example, while the underlying simulation may have an integration rate of 100 Hz, a guidance algorithm may be called at 1 Hz and an attitude controller at 20 Hz. Note that this interface does not operate in real time, and these call frequencies occur within simulation time.

The DQ-ADMM algorithm was wrapped with a C interface that assembled the required vehicle configuration, states, and user-defined constraints, and POST2 truth data. In the case of the DQ-ADMM algorithm, the guidance was called once at the guidance initialization event and generated a single reference trajectory in the form of vehicle states, aerodynamic forces, and vehicle body moments at each node. This data was then transformed into planet-relative angle of attack (α), sideslip (β), and bank angle (σ) commands that were then interpolated and fed to the POST2 3DOF steering model.

B. Implementation Verification

To verify the C++ implementation, the translated C++ DQ-ADMM code was compiled as a standalone executable and run with identical inputs as the original provided MATLAB code. The results of the standalone C++ program was compared to that of its MATLAB equivalent to provide confidence in the accurate translation between the two programming languages. This process also aided in the debugging efforts when translating to the C++ codebase and provided a benchmark of expected differences between the original MATLAB delivery and Armadillo computations. Fig. 2 shows the position comparison between the two implementations while Fig. 3 illustrates the flight path, heading, and bank angle comparisons. Agreement between the two implementations was not numerically identical but was an expected result due to use of different libraries and numerical integration methods. Disagreement in the values at the initial node are noted, however, the level of agreement was deemed appropriate to continue with implementation in POST2.

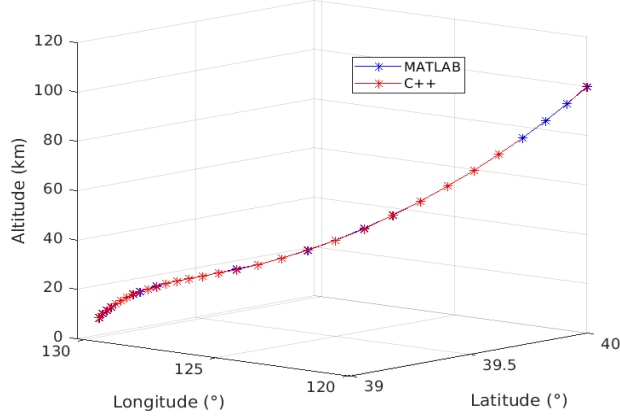


Fig. 2 Spacecraft position comparison of original MATLAB and C++ implementations of DQ-ADMM.

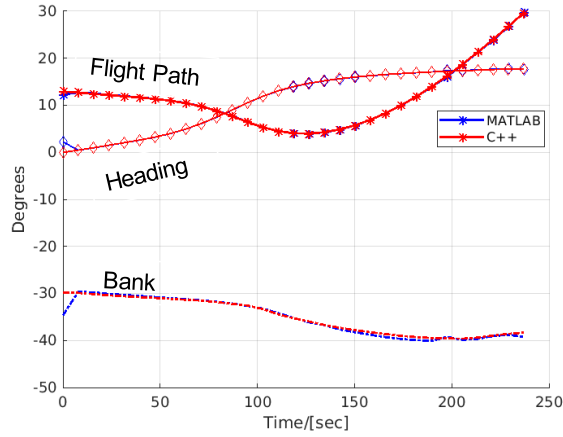


Fig. 3 Spacecraft flight path, heading, and bank angle comparison of original MATLAB and C++ implementations of DQ-ADMM.

C. Integration Challenges

Due to the complex nature of the DQ-ADMM guidance algorithm, various challenges were encountered in its implementation into the POST2 simulation framework. The majority of these challenges involved the verification of the C++ matrix math operations applied to matrices of size $[3,000 \times 3,000]$ compared to the corresponding results from MATLAB. Small differences in states could result in differences in matrix elements that were not readily diagnosable. This challenge of diagnosing differences in matrix elements is partly due to C++ debuggers not having the same utility MATLAB workspaces offer when trying to evaluate large matrices mid-program execution. Lessons learned from this approach include greater reliance on unit testing and more direct collaboration with the code authors.

D. Runtime Considerations

Simulation runtime, or wall time, is a significant factor when conducting statistical analysis via methods such as Monte Carlo. It can be a computational challenge when conducting a typical Monte Carlo analysis that may include 8,000 or more sample trajectories. Table 1 lists the runtimes of the different implementation. The higher runtimes of the C++ implementation are noted and attributed primarily to not focusing on code optimization. Additionally, perfect agreement is not expected as these two cases shown in the table do not use identical inputs. The POST2 case uses data from the simulated NASA reference trajectory. The C++ implementation is considered sufficiently fast to run a Monte Carlo analysis.

Table 1 DQ-ADMM wall times.

Implementation	ADMM Iterations	ADMM Wall Time (s)	Total Wall Time (s)
MATLAB	126	129.92	284.80
POST2 / C++	124	137.70	303.67

IV. Application: Human-Scale Mars EDL Architecture

The POST2 simulation used to evaluate the DQ-ADMM guidance algorithm was the NASA human-scale low L/D Mars entry system that has been used in a variety of NASA analyses [4, 5, 20, 21]. The EDL concept of operations is shown in Fig. 4 (note that this work is focused on entry guidance and powered descent is not modeled). The spacecraft is modeled as a Hypersonic Inflatable Aerodynamic Decelerator (HIAD) with an entry mass of approximately 47.5 t. The DQ-ADMM guidance is activated at approximately 100 km altitude where the atmosphere becomes sensible enough to affect a vehicle's dynamics. At this stage, the only forces acting on the spacecraft are gravitational and aerodynamic.

For this work, two modifications were made to the simulation to facilitate comparisons with the inner loop trajectory. First, the simulation atmosphere, originally modeled with MarsGRAM, was replaced with a lookup table based on a similar polynomial curve fit used in the guidance using a generic and public Mars atmosphere model [22]. Note that the inner loop polynomials use normalized coefficients. Second, the vehicle aerodynamics model was replaced with a lookup table based on the polynomial curve fit used in the guidance in which the aerodynamic coefficients are functions of sideslip and angle of attack. Additionally, the version of the simulation used in this work was 3DOF to facilitate integration and verification of guidance command pass-through. Forward work will include replacing these truth models with their original versions used in other work and switching to the 6DOF simulation used in integrated performance assessments [5].

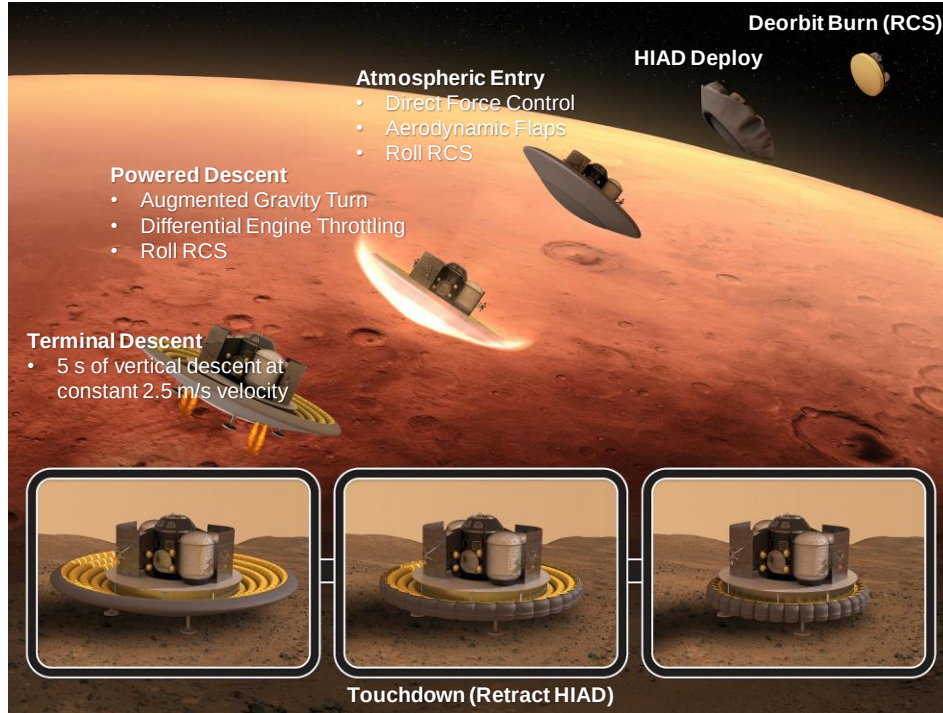


Fig. 4 Entry vehicle concept of operations. Note that powered descent and landing are not modeled in this work.

V. Results

The results of applying the DQ-ADMM algorithm in POST2 to the 3DOF human-scale low L/D entry system is discussed in this section. Parameters are plotted as a function of altitude rather than the typical time to highlight the known differences between the inner and outer loop density profiles.

Fig. 5 shows the commanded and actual aerodynamic angles as a function of altitude. Recall that these commands are derived from the DQ-ADMM algorithm outputs and a major challenge in implementation involved ensuring accuracy in coordinate frame definitions and transformations between the unique frames used in the DQ-ADMM and POST2. In the figure, it can be observed that the angle of attack (α) profile is maintained between a reasonable $\pm 5^\circ$, and the angle of sideslip (β) is similarly reasonable between 3° and -10° . The magnitude of the bank angle (σ) remains below 25° indicating a constant lift up trajectory.

Constraints of $\pm 26^\circ$ were imposed on the spacecraft's angle of attack and sideslip angle. These constraints were adapted from previous EDLAS aerosciences studies that analyzed HIAD systems for EDL missions and was communicated to the ECF awardees. This attitude constraint is imposed to avoid flow impingement and radiative heating effects on the payload at high angles of attack.

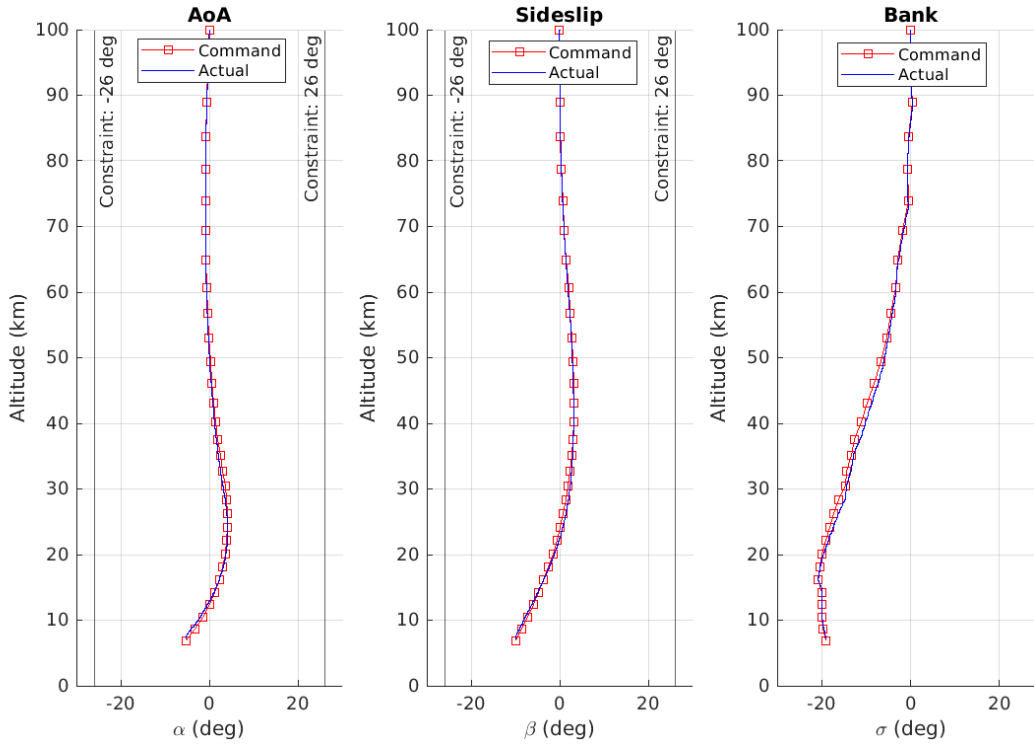


Fig. 5 Commanded and actual aerodynamic angles.

Fig. 6 shows the predicted (commanded) and actual aerodynamic forces acting on the entry vehicle. While there is positive initial agreement, the forces quickly diverge, indicating a mismatching either between the inner and outer loops, or the truth vehicle unable to match the commanded aerodynamic angles. Observed differences between the predicted and actual aerodynamic forces in Fig 6 are partially attributed to differences in atmosphere modeling and integration schemes utilized between the inner and outer loop trajectory.

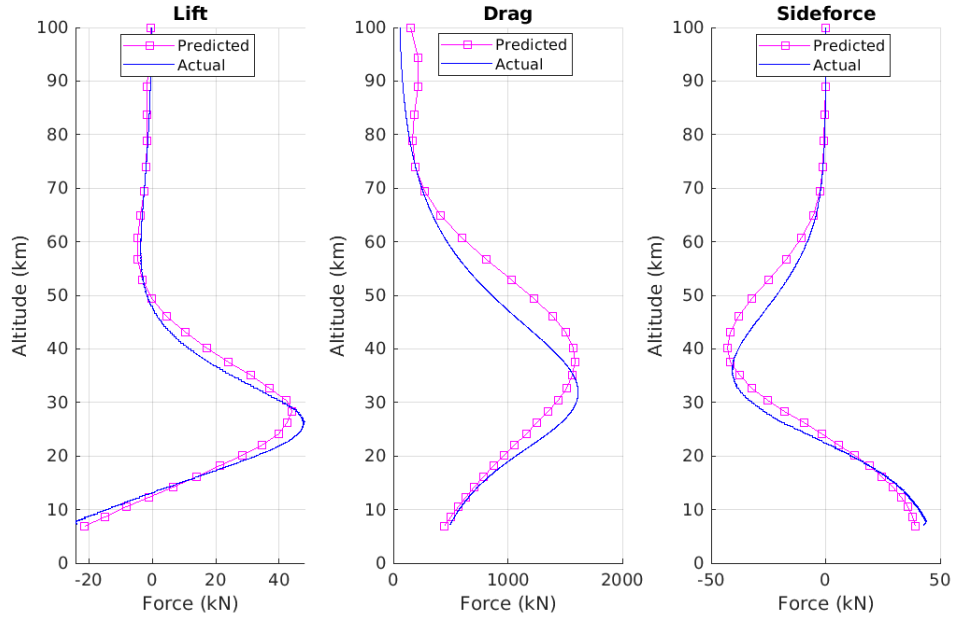


Fig. 6 Predicted and actual aerodynamic forces.

Fig. 7 illustrates the different density profiles observed between the inner (predicted) and outer (actual) loop trajectories. The predicted density is derived from the dynamic pressure output from the guidance. The density modeled in the predicted trajectory is significantly denser when compared to the actual and is the primary driver on where the peak aerodynamic forces occur with respect to altitude in Fig. 6. Simulations utilizing different density profiles can result in significantly different trajectories and performance.

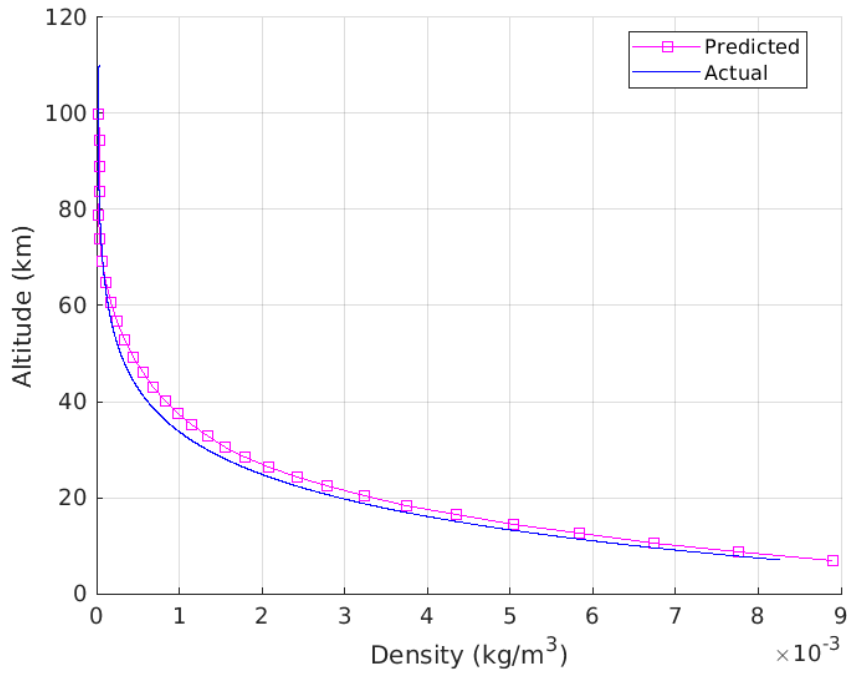


Fig. 7 Predicted and actual density.

Fig. 8 illustrates the operational constraints placed on the structural loading expressed in Earth g's and convective heat flux evaluated at the spacecraft's stagnation point. The maximum g's for the actual and predicted trajectories were evaluated to be 3.37 g and 3.31 g, respectively, falling below the imposed 5 g constraint. The heat flux is calculated using the Sutton-Graves equation for stagnation-point convective heating. The maximum calculated heat fluxes were evaluated to be 7.99 W/cm² and 8.57 W/cm² for the actual and predicted trajectories, respectively. Although the predicted heat flux surpasses the 8.0 W/cm² constraint threshold, its error is within the user defined tolerance mentioned in Section II. The significant differences where peak g's and heat fluxes occur with respect to altitude are again emphasized by the profile in Fig. 7.

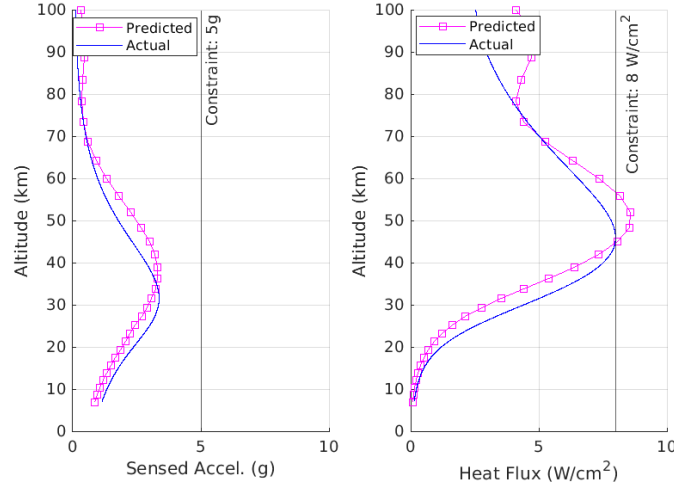


Fig. 8 Predicted and actual g's and heat flux.

Last, the most significant differences observed between the predicted and actual trajectories are seen in Fig. 9 which shows the spacecraft's position with respect to the planet-centered relative frame. The predicted and actual positions quickly diverge resulting in a difference of 62.7 km at simulation termination. It is important to note that there was no imposed constraint on final latitude or longitude, however a final altitude constraint of 7.0 km was implemented in which both trajectories achieved. These position differences are attributed to unidentified discrepancies between the coordinate frames used in the inner and outer loop simulations and differences in numerical integration schemes.

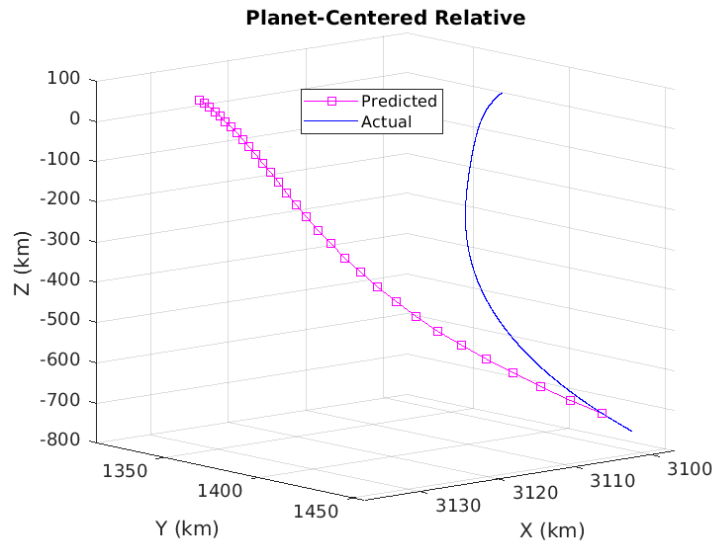


Fig. 9 Predicted and actual position near end of entry phase.

VI. Summary and Conclusions

The DQ-ADMM guidance algorithm described in [8, 9] was translated to C++ and implemented in the POST2 simulation framework. The C++ guidance was verified using the original MATLAB code, and while results were not identical, the differences were small enough to justify continuing implementation. Lessons learned from the implementation included more investment in unit and system level testing, and more verification effort allocated to coordinate frames and guidance assumptions. The translated code was applied to a NASA reference Mars entry system [4] and simulated as-flown performance was compared against the DQ-ADMM predicted performance. While inner and outer loop model differences were noted and resulted in trajectory divergence, the results were promising, and further work is expected to resolve these differences.

Acknowledgments

The authors thank Dr. Ran Dai of Purdue University for developing and providing the original MATLAB version of the guidance code, and Dr. Charlie Wan of Tuskegee University for valuable information and further discussion of the algorithm. The authors also thank Hayden Dean and Natalia Marchini of the Georgia Institute of Technology for implementation support of the DQ-ADMM guidance code in POST2.

The work presented in this paper was funded in part by the NASA Space Technology Mission Directorate (STMD).

References

- [1] Dwyer-Cianciolo, A. M., and Polsgrove, T., “Human Mars Entry, Descent and Landing Architecture Study Overview,” AIAA 2016-5494.
- [2] Dwyer-Cianciolo, A. M., and Polsgrove, T., “Human Mars Entry, Descent, and Landing Architecture Study: Descent Systems,” AIAA 2018-5190.
- [3] Percy, T. K., Polsgrove, T., Sutherlin, S., and Dwyer-Cianciolo, A. M., “Human Mars Entry, Descent, and Landing Architecture Study: Phase 2 Summary,” AIAA 2018-5193.
- [4] Dwyer-Cianciolo, A. M., Korzun, A. M., Edquist, K., Samareh, J., Sostaric, R., Calderon, D., and Garcia, J. A., “Human Mars Entry, Descent, and Landing Architecture Study: Phase 3 Summary,” AIAA 2020-1509.
- [5] Lugo, R. A., Powell, R. W., and Dwyer-Cianciolo, A. M., “Overview of a Generalized Numerical Predictor-Corrector Targeting Guidance with Application to Human-Scale Mars Entry, Descent, and Landing,” AIAA 2020-0846.
- [6] “Early Career Faculty (ECF) 2018,” <https://www.nasa.gov/directorates/stmd/space-tech-research-grants/early-career-faculty-ecf-2018/>, accessed Nov. 1, 2024.
- [7] “Optimized Entry and Powered Descent Guidance for Precision Planetary Landing,” <https://www.nasa.gov/directorates/stmd/space-tech-research-grants/optimized-entry-and-powered-descent-guidance-for-precision-planetary-landing/>, accessed Nov. 1, 2024.
- [8] Wan, C., Jing, G., Dai, R., and Rea, J. R., “Fuel-Optimal Guidance for End-to-End Human-Mars Entry, Powered-Descent, and Landing Mission,” AIAA 2020-1472.
- [9] Wan, C., Pei, C., Dai, R., Jing, G., and Rea, J. R., “Six Dimensional Atmosphere Entry Guidance based on Dual Quaternion,” AIAA 2021-0507.
- [10] Reynolds, T. P., Szmuk, M., Malyuta, D., Mesbahi, M., Acikmese, B., and Carson, J. M., “Dual Quaternion-Based Powered Descent Guidance with State-Triggered Constraints,” AIAA Journal of Guidance, Control, and Dynamics, Vol. 43, No. 9, September 2020.
- [11] Kamath, A. G., et al., “Customized Real-Time First-Order Methods for Onboard Dual Quaternion-based 6-DoF Powered Descent Guidance,” AIAA 2023-2003.
- [12] Doll, J. A., et al., “Performance Analysis of a Dual Quaternion Guidance Algorithm applicable during Lunar Approach with a Hazard Avoidance Maneuver,” AIAA 2024-1757.
- [13] Shaffer, J., et al., “Implementation and Testing of Convex Optimization-based Guidance for Hazard Detection and Avoidance on a Lunar Lander,” AIAA 2024-1584.
- [14] Williams, R. A., Lugo, R. A., Marsh, S. M., Hoffman, J. A., Shidner, J. D., and Aguirre, J. T., “Enabling Thread Safety and Parallelism in the Program to Optimize Simulated Trajectories II,” AIAA 2023-0148.
- [15] Braun, R. D., et al., “Mars Pathfinder Six-Degree-of-Freedom Entry Analysis,” JSR Vol. 32, No. 6.
- [16] Striepe, S. A., “Mars Science Laboratory Simulations for Entry, Descent, and Landing,” JSR Vol 43, No. 2 pp.311-323.
- [17] Dutta, S., et al., “DAVINCI Venus Entry, Descent, and Landing Modeling and Simulation,” AIAA 2023-1165.
- [18] Sanderson, C. and Curtin, R., “Armadillo: A Template-Based C++ Library for Linear Algebra,” Journal of Open Source Software, Vol. 1, No. 2, pp. 26, 201.
- [19] Sanderson, C. and Curtin, R., “Practical Sparse Matrices in C++ with Hybrid Storage and Template-Based Expression Optimisation,” Mathematical and Computational Applications, Vol. 24, No. 3, 2019.
- [20] Lugo, R. A., Karlgaard, C. D., Powell, R. W., and Dwyer-Cianciolo, A. M., “Integrated Flush Air Data Sensing System Modeling for Planetary Entry Guidance with Direct Force Control,” AIAA 2019-0663.

- [21] Lugo, R. A., Dwyer-Cianciolo, A. M., Dutta, S., Williams, R. A., Pensado, A., and Chen, P., “Integrated Precision Landing Performance Results for a Human-Scale Mars Landing System,” AIAA 2022-0607.
- [22] “Mars Atmosphere Equation – English,” <https://www1.grc.nasa.gov/beginners-guide-to-aeronautics/mars-atmosphere-equation-english/>, accessed 16 September 2024.