Space Launch System Liftoff and Separation Dynamics Analysis Tool Chain

Benjamin S. Burger*, Carole Addona†, Benjamin Diedrich‡, William J. Harlin§, Peter McDonough**, Zachary Muscha††, Ray Sells‡‡, Daniel Tyler§§

A flexible, hierarchical tool chain that is being applied to NASA's Space Launch System (SLS) for critical dynamics phenomena is described. This tool chain, called CLVTOPS, is used to investigate lateral liftoff movement of the vehicle as it departs and clears the mobile launch tower and separation of the two solid rocket boosters without collision with the core stage and payload. The toolset's architecture was configured to take advantage of a modern software-engineering approach for maximum flexibility and utilization of open-source simulations and associated tools. As opposed to a "monolithic" approach, scripting languages were used to "bind" together a tool chain to configure and organize input data, execute and produce analysis results, and post-process these results to facilitate a rapid iterative analysis process to quickly address issues and pursue alternatives with emphasis on analysis automation. Key capabilities in the tool chain include processing and mining of very large data sets, a wide range of graphical depictions, and high-fidelity, physics-based simulations. The paper begins with a problem description and the motivation for liftoff and separation dynamics analysis followed by a historical survey of dynamics analyses for previous NASA human-rated launch vehicles. Details of the tool chain and its components are then introduced divided, first, into description of the scripting language architecture used to "bind" the simulation tools, programs, and scripts together and, second, the physics models and simulations. Representative analyses and data products are shown for liftoff and booster separation dynamics that provide in-depth insight to the tool chain's capabilities. Supporting activities such as simulation tool chain verification, version archiving and data management, and training are addressed. The paper concludes with case examples on how the tool chain can be tailored to related aerospace dynamics analyses, both large and small. These patterns and techniques for SLS dynamics tool construction can be applied for other aerospace simulations.

I. Introduction

The National Aeronautics and Space Administration's (NASA) Artemis Program, utilizing the Space Launch System (SLS), will enable human access to space at an unprecedented scale, providing exploration-class mission capability to return humans to the moon and beyond. To that end, NASA's development of the SLS family of launch vehicles employs many new capabilities and responds to numerous challenges in the flight and operation of this large-scale system. A top-level diagram of the initial Block-1 version of SLS is shown in Figure 1.

The Artemis Program presents numerous unique challenges for SLS; although analogies can be drawn to the highly successful Apollo Program[1], SLS is charged to deliver "boots on the moon" with considerably fewer flights[2], which places extreme emphasis on robust and accurate ground testing and simulation. In addition, the reduced timeline given to accomplish the moon landing goal[3] dictates rapid design and analysis iterations to refine and qualify a dependable vehicle. These distinctive challenges present an extraordinary opportunity to explore and utilize the full power of remarkable advances in computer hardware computation capabilities and simulation technology over the past several decades.

For the first flight, Artemis I, two key SLS vehicle dynamics phenomena must be analyzed to establish confidence in the vehicle and certify it for launch. First, the liftoff movement of the vehicle as it separates from the launch pad, departs and clears the Mobile Launcher (ML) tower and umbilicals, and clears the proximity of the Lightening Protection System (LPS) must be shown to avoid any detrimental collisions and undue damage to the launch pad.

* SLS Liftoff and Separation Dynamics Lead, EV42 Guidance, Navigation, and Mission Analysis Branch, Jacobs Space Exploration Group (JSEG)/Dynamic Concepts, Inc., Huntsville, AL 35806
† Aerospace Engineer, EV42 Guidance, Navigation, and Mission Analysis Branch, JSEG/Engineering Research and Consulting, Inc., Huntsville, AL 35806
‡ Aerospace Engineer, EV42 Guidance, Navigation, and Mission Analysis Branch, JSEG/Dynamic Concepts, Inc., Huntsville, AL 35806
§ SLS Liftoff and Separation Dynamics Deputy Lead, EV42 Guidance, Navigation, and Mission Analysis Branch, JSEG/Dynetics, a Leidos Company, Huntsville, AL 35806
** PhD, Aerospace Engineer, EV42 Guidance, Navigation, and Mission Analysis Branch, JSEG/Qualis Corporation, Huntsville, AL 35806
†† Aerospace Engineer, EV42 Guidance, Navigation, and Mission Analysis Branch, JSEG/Jacobs, Huntsville, AL 35806
‡‡ Aerospace Engineer, EV42 Guidance, Navigation, and Mission Analysis Branch, JSEG/DESE Research, Inc, Huntsville, AL 35805
§§ Aerospace Engineer, EV42 Guidance, Navigation, and Mission Analysis Branch, NASA/MSFC, Huntsville, AL 35812

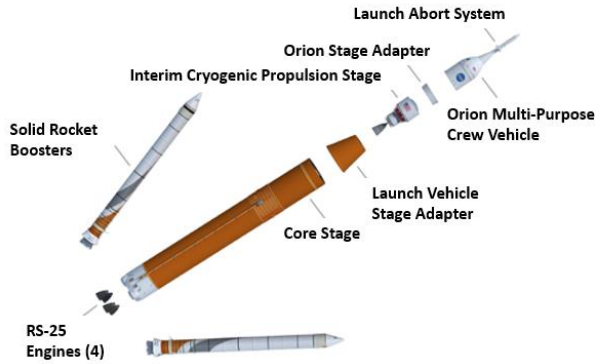


Figure 1. SLS Block-1 Layout. *Shown here is the crewed configuration; a cargo configuration replaces the Multi-Purpose Crew Vehicle with a payload compartment.*

Second, later in flight well before main-engine cut-off, the two Solid Rocket Boosters (SRBs) must separate without collision, or recontact, with the Core Stage (CS) and payload. Analyses must be of high enough fidelity and resolution to warrant a high degree of confidence that these critical SLS flight segments can be successfully executed.

Dynamics analyses will also be performed to cover additional separation events of the planned Block-1 cargo and Block-1B variants of SLS. These separation events include the separation of the CS from the Exploration Upper Stage (EUS), the separation of the Universal Stage Adapter (USA) from the EUS, the separation of the Payload Fairing (PLF) panels and the separation of payloads from SLS. The main focus of this paper will be a description of the analysis tool chain used to perform an evaluation of launch vehicle dynamics for the Block-1 liftoff and booster separation events.

The Liftoff and Separation Dynamics team from NASA's Marshall Space Flight Center (MSFC) has developed an integrated tool chain analysis capability to quickly investigate these SLS launch dynamics requirements and issues. This unique linking-of-tools provides NASA senior leadership timely information in an iterative development context while adhering to challenging schedule requirements. The toolset's architecture was configured to take advantage of a modern software engineering approach for maximum flexibility and utilization of "best-of-breed" simulations and associated tools. As opposed to a "monolithic" approach, scripting languages were used to "bind" together a tool chain to configure and organize input data, execute and produce analysis results, and post-process these results to communicate relevant outcomes to decision makers, as well as provide direction for further design improvements or more in-depth analyses. Key capabilities in the tool chain include processing and mining of very large data sets, a wide range of graphical depictions, and high-fidelity, physics-based simulations. Automation in executing the tools and analyses is an integral component to the requirement for quick turnaround. The goal is to provide decision makers a comprehensive, but hierarchically organized, data set and the ability to iterate quickly and efficiently through many cycles of the question and answer process that drives analysis.

In order to provide an overview and introduction to this dynamics tool chain, the paper begins with a problem description and the motivation for launch and separation dynamics analysis. Of course, the launch and separation phenomena addressed here are not unique to SLS and have been addressed before, notably in other NASA human-rated launch programs: Apollo, Space Shuttle, and Constellation. The origin of this type of analysis for past large-scale human-rated launch systems is described. This historical background includes how the analysis methodology and technology advanced along the way as more knowledge was gained and computational ability increased. Details of the tool chain and its components are then provided. First, a description of the philosophy and architecture used to "bind" the simulation tools, programs, and scripts together is introduced. The application of scripting languages to preprocessing, data file configuration, execution, and post-processing in the tool chain is described. The launch dynamics problems of clearing the ground structures and, later on in flight, separation of the boosters is examined in depth. Analysis methodology and reporting is shown, primarily with representative data products. These data products emphasize visualization and distillation of statistical results to actionable measures of performance. A section is included on dynamics analysis support activities such as simulation tool chain verification, version archiving and data management, and training. The paper concludes with case examples on how the tool chain can be tailored to related aerospace dynamics analyses, both large and small.

Figure 2. SLS Tower Complex and LPS. *Artist's rendition of the SLS vehicle stack on the physical ML inside the LPS.*

II. Problem Description and Motivation for Analysis

In-depth dynamics analysis is required to assure successful liftoff from the ground structures and booster separation from the CS during flight. These analyses are addressed separately.

A. Liftoff Dynamics and Ground Systems Clearance

The tower complex and associated LPS are shown in Figure 2. A keep-out zone (KOZ) is defined as a volume around the ground system hardware that the SLS vehicle is not allowed to enter during liftoff, or re-occupy once it has exited. To expedite analysis and provide a degree of margin, the KOZ is enforced by a simplified outer mold line (OML) volume, shown in Figure 3. Analysis is needed to provide information on the motion, clearances, and control performance of the vehicle in the proximity of ground systems, e.g. Tower, Tail Service Mast (TSM), Vehicle Support Posts (VSP), ML deck, and the LPS. The minimum separation distance (clearance) between any portion of the vehicle and the KOZ is the primary metric to be considered for liftoff analysis. The liftoff clearance design objective is a 99.865 percentile with 90 percent confidence level (CL) probability that the vehicle will not intrude into the static KOZ during liftoff.

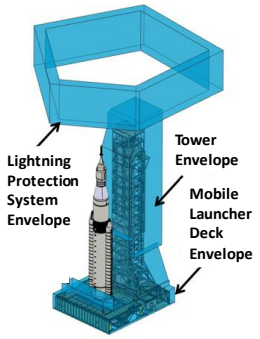


Figure 3. SLS Tower and LPS Keep Out Zones.

B. Booster Separation

A pair of SRBs, depicted in Figure 1, are used in parallel with the CS of the vehicle during the early portions of ascent flight. The boosters provide the majority of the required thrust to lift the combined vehicle stack. When they are near burnout, forward and aft attach points are severed, Booster Separation Motors (BSM) are fired, and the boosters are detached allowing the CS to continue to propel the vehicle to orbit without the mass of now depleted SRBs.

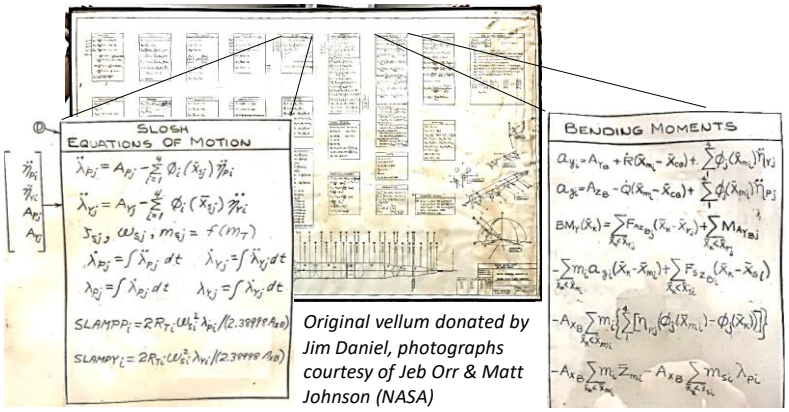
The design objective for this separation event is a 99.865 percentile with 90 percent CL probability that the boosters will separate without re-contacting the CS or any other part of the launch vehicle during the separation event. The minimum separation distance (clearance) between any part of the CS and boosters is one of the primary measures to be considered for separation performance, flight safety, and reliability. Booster separation centerline drift envelopes, minimum separation distance versus time curves, and vehicle/booster state versus time plots are key measures of performance for showing expected nominal and off-nominal separation execution, as well as sensitivities to various performance drivers.

III. Past Work

The tool chain constructed for SLS benefits from a heritage of preceding NASA human-rated launch vehicle experience and analyses.

A. Apollo

It was recognized very early on that launch vehicle interactions with the launch complex needed to be fully addressed. The Apollo Program's Saturn V was the first very large launch system built by the United States. A true monster of a launch vehicle, it generated over 33 million newtons of thrust at liftoff and carried 2.5 million kilograms of fuel and oxidizer. In a malfunction scenario, the Saturn V could explode with the force of a small atomic bomb[4]. It was immediately recognized that the most likely cause of an on-pad explosion of a Saturn V was a collision with the tower during liftoff.



Original vellum donated by Jim Daniel, photographs courtesy of Jeb Orr & Matt Johnson (NASA)

Figure 4. Original Saturn V Flight Dynamics Models.

In 1964, NASA conducted a study to determine what factors could initiate a tower collision. The leader of this study, Mowery[4], considered seven factors that could disturb the liftoff path of the vehicle. These were: a variation in the holddown force of plus or minus 15%, a variation in thrust of 4%, engine misalignment, an offset in the vehicle's center of gravity, wind, engine failure, and an "engine hardover." Mowery's analysis concluded that: "no problem as to tower collision exists for combined disturbances if a malfunction does not occur." In other words, wind or a misaligned vehicle, even in combination, could not cause a tower collision in the absence of a malfunction. Additionally, neither an engine failure nor an engine hardover for either of the inboard engines could cause a tower

collision in the absence of a malfunction. Even at this early time, state-of-the-art modeling and simulation was advanced enough to consider the effects of flexible body bending and propellant slosh in high-fidelity, time-based simulation[5]. Figure 4 shows close-up digital photographs from the original vellum flight dynamics models for flex and slosh from 1966.

Similar early work was done to address the stage separation dynamics. Wasko[6] experimentally investigated stage separation dynamics at the NASA Lewis supersonic wind tunnel. Chubb[7] was one of the first to address stage separation for the Saturn vehicle. He focused on the collision boundary between two separating stages of the Saturn-Apollo-4 vehicle. Work in this area (see Decker[8,9], for instance) progressed to parallel-staged vehicles that would eventually be applied to the Space Shuttle.

NASA subsequently launched thirteen Saturn V's without a single catastrophic failure and, focusing on the launch and separation dynamics, no detrimental recontacts during liftoff and stage separation.

B. Space Shuttle

Post-Apollo, NASA required a more advanced launch vehicle dynamics analysis capability to address the added geometric complexity of the Space Shuttle, which consisted of an orbiter, an external tank (ET), and two side-mounted SRBs, as well as the complex dynamics problem of staging parallel boosters at supersonic speeds from an actively thrusting, winged, and crewed vehicle. This more complex configuration presented different separation systems to analyze: boosters would first separate from the orbiter/ET and, later in flight, the orbiter would separate from the ET. Flight dynamics predictions, component malfunction assessment, and possible recontacts of the separating boosters were all areas of analysis reviewed to ensure the design was sufficient to accomplish each mission's goals. Supersonic staging of parallel SRBs was complicated by solid rocket motor thrust mismatch and complex aerodynamics.

For liftoff, the primary analytical tool was the 'M50'[10] six-degree-of-freedom (6DOF) computer simulation. Outputs from M50 were used by a separate post-processing tool to determine liftoff clearances. These included clearances of the SRB nozzle-to-holddown support posts, orbiter wing tip-to-lightening mast, and SRB aft skirt-to-blast deflection shields, among many. The 6DOF model was anchored to photographic data from cantilever dynamics testing performed while the launch vehicle was attached to the pad in order to predict vehicle excursions prior to launch. Also, time histories of dynamics and flight control parameters were compared to simulation during launch. Worst-case clearance predictions were generated from M50 and subsequently validated by very good comparison between liftoff displacements and those derived from photographic data.

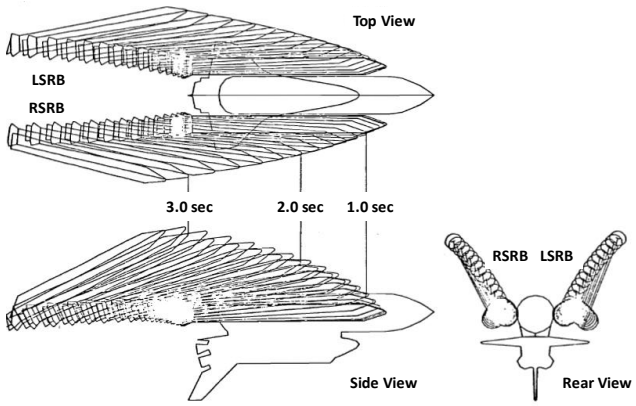


Figure 5. Three View Strobe Plots. *Example of simulation output from the Shuttle Program predicting how the SRBs would travel after the separation event.*

For SRB separation, the Space Vehicle Dynamics Simulation (SVDS) was a 6DOF, high-fidelity separation dynamics simulation used as the primary off-line simulation tool[11]. Verification of the simulation math models was performed by comparing measurements taken during initial operational test flights with SVDS predictions. Figure 5 is an example of one of these predictions, depicting the predicted path of the SRBs after they fell away from the orbiter/ET stack. The SRBs were required to follow a distinct "upward" trajectory in order to avoid contacting the protruding orbiter wings.

The accuracy and value of these analyses were validated throughout the Space Shuttle Program where all observed liftoff and booster separation events were nominal. These analyses also provided a strong template for current SLS separation analyses.

C. Constellation

Ares I was the first planned launch vehicle of NASA's Constellation Program, the human-rated system meant to follow the Space Shuttle Program. Ares I was a single stack configuration with a first stage booster and J-2X powered upper stage. The main dynamic modeling tool described later in this report, CLVTOPS[***], had its origins with this vehicle analysis[12]. It was at this point that the launch vehicle liftoff and separation analyses were sub-divided into

[***] CLVTOPS is the proper name of the tool and not an acronym

distinct components to capture each separation event: liftoff, first stage/upper stage separation, Encapsulated Service Module (ESM) panel jettison, and Ullage Settling Motor (USM) separation. It was recognized that since all of the liftoff and separation events shared common input models and source code (such as mass properties, propulsion, aerodynamics, and flight software), it would be efficient to model all of the events using a single, common simulation.

A 6DOF multi-body time domain flight dynamics tool was used for tower clearance analysis and separation analysis. The separation events were modeled using 6DOF hinges between the separating bodies. Prior to separation, the hinges employed high translational and rotational stiffness and damping to constrain the bodies together. To simulate the physical separation events, the stiffness and damping values were set to zero, essentially disconnecting the two bodies. This modeling paradigm was to become the standard way to model separating bodies for subsequent SLS simulations.

Excellent comparisons with Ares I-X flight data provided quantitative evidence for validation of this simulation modeling methodology[12].

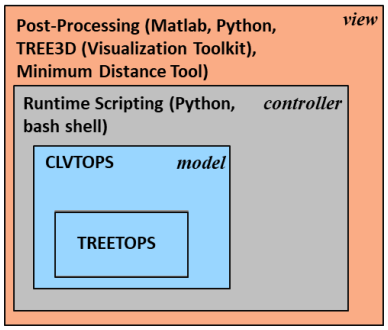


Figure 6. Tool Chain Hierarchical Composition. *Note use of scripting languages to "wrap" the core TREETOPS dynamic computational engine.*

IV. Launch and Separation Dynamics Analysis System Overview and Architecture

As noted in the previous section, configuration of what was to become the SLS launch dynamics tool chain originated during the Constellation Program. A "wrapper" approach consisting of many tools tied together with scripting languages evolved and became the de facto architecture. The tool chain is structured and best portrayed as a hierarchical nested view of each of the tools as shown in Figure 6. Table 1 is a brief description of each component. An important aspect of every component is that each scripting language is open-source with the exception of MATLAB®, which is a widely used, industry standard tool. This open-source feature maximizes the flexibility in configuring individual codes, interfacing them together, and having complete control over their modification, utilization, and execution. Following is a description of each tool in the hierarchy and their relationship to each other starting with the most "inward" tool, TREETOPS[†††]. This hierarchy is roughly analogous to the execution sequence as it proceeds from physics model execution, and progressing "outward", to output data processing and presentation of results.

Table 1. List of Code Components.

Code or Tool	Description
TREETOPS	Generic time-history simulation tool for complex multi-body flexible structures
CLVTOPS	Wrapper around TREETOPS to add launch–vehicle-, including SLS-, specific models
Minimum Distance Tool	Post-processing tool that uses triangular mesh models and position and orientation data of vehicle components to compute clearances during liftoff and separation events
Python	Dynamically-typed scripting language used in run-time scripting of CLVTOPS, Monte-Carlo runs, and post-processing tasks
Bash shell	Operating system-level scripting shell
NumPy	Python scientific computing extensions including array objects and linear algebra functions
TREE3D	Python tool developed with the Visualization Toolkit (VTK) to animate the simulation results
Matplotlib	Python library used for visualizing post-processing data
MATLAB®	Commercial software used for post-processing data and visualization

A. TREETOPS

The core code is TREETOPS[13], which was originally developed under NASA funding. TREETOPS is a generic time history simulation tool developed for analysis of the dynamics and control-related issues of complex multi-body flexible structures with active control elements. The name "TREETOPS" is indicative of the tree topology of linked multiple bodies, each of which may be rigid or flexible, with translations and large angle rotations between each body. Kane's method[14] is employed to synthesize the equations of motion, which are numerically integrated, to generate the time history response of the system. One attractive feature of Kane's method is that the equations of motion for a

[†††]TREETOPS is the proper name of the tool and not an acronym

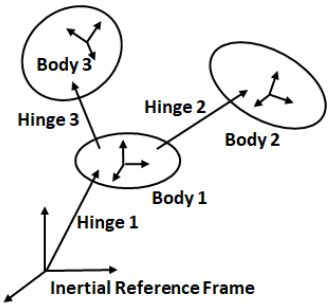


Figure 7. Multi-Body Representation in TREETOPS. *Bodies within TREETOPS always start with a hinge connection from the inertial reference frame to the first body. Hinge 1 implicitly defines the inertial frame of the simulation.*

system are very systematically formulated, making it easy to automate with computer code. Kane's equations give a substantial runtime advantage versus Newtonian formulations because the constraint forces between bodies do not need to be included in the equations of motion, reducing the number of algebraic differential equations which need to be solved[15]. Figure 7 illustrates the abstract hinge-body tree topology representation of a system within TREETOPS. Table 2 lists some prominent NASA MSFC projects where TREETOPS was used to analyze rather complex interactions between multi-body flexible structures and control systems.

Several improvements have since been made to TREETOPS. A major enhancement was added to significantly improve runtime. Kane's method is a powerful algorithm that reduces the multi-body dynamics (i.e. code) into "generalized partial velocities". However, these derivatives do not appear explicitly, so it is necessary to invert a matrix so that the derivative terms can be integrated[26]. It was recognized that the matrix inversion was only an intermediate step and not a product of the analysis. Also it was noted that Kane's method ensures that the matrix to be inverted is always positive-definite symmetric[27]. Therefore, a Cholesky decomposition of the matrix could be used and simple back-substitution could be applied to define the generalized velocity derivatives explicitly. This was implemented in TREETOPS using the Linear Algebra PACKage/Basic Linear Algebra Package (LAPACK/BLAS)[28] standard numerical library as optimized in the Intel Math Kernel Library®[29] routines. Runtime reductions on the order of 30-50% were typically observed, with the reduction increasing exponentially with the number of states. This runtime reduction was particularly valuable where a large number of flex modes (i.e. states) were present in the system. Another significant TREETOPS improvement was undertaking migration of code from Fortran77 (the original code) to Fortran 90 to leverage additional functionality and code readability. Finally, a generic surface-to-surface contact model was added to TREETOPS. This model is necessary when modeling separation events where the separating hardware is allowed to either rub or have incidental recontact. The model can also be used to predict recontact dynamics and loads, which could potentially be used in structural analysis to assess the severity of undesired recontact events.

Table 2. TREETOPS Projects at Marshall Space Flight Center.

Project
Space Shuttle Pinhole Occulter[16]
Space Shuttle Astro Observatory[17]
Hubble Space Telescope[18]
International Space Station[19] [20] [21] [22]
Chandra X-ray Observatory[23]
Gravity Probe B[24]
Solar sail spacecraft[25]

B. CLVTOPS

TREETOPS provides the underlying computational physics engine for the next layer in the hierarchy, CLVTOPS. CLVTOPS is the central simulation for the tool chain and, hence, the tool chain itself is commonly referred to as CLVTOPS.

The CLVTOPS 6DOF simulation performs the computations for SLS's launch and separation dynamics. As noted earlier, CLVTOPS was originally created by NASA MSFC in the early 2000's for the Constellation Program[12] and has evolved with many improvements including better modularization, leverage of Fortran 90 features, and built-in interfaces and commonality with external pre- and post-processing scripts. Also, the number of and depth of supporting physics models were greatly expanded.

CLVTOPS interfaces to TREETOPS as a wrapper where CLVTOPS code gets called as part of every TREETOPS integration cycle. At a top-level sense, CLVTOPS computes the forces and moments that feed into the TREETOPS dynamics engine. Also, since the TREETOPS formulation assumes constant mass properties, CLVTOPS computes the updated launch vehicle mass properties as propellant is consumed, and overwrites the mass properties used in the generalized speed derivatives for TREETOPS. TREETOPS takes care of propagation in time, vis-a-vis integration of the derivatives. The interface is two-way as the integrated states are passed back to CLVTOPS to define the forcing functions for the next cycle. Specific models in CLVTOPS that support computation and reporting the forcing functions are listed in Table 3. Table 4 highlights some more in-depth features of these CLVTOPS models and functions that provide a high degree of fidelity. A key design goal of CLVTOPS was to leverage any existing software to aid computations. For instance, the SPICE library[30] was added as a component of CLVTOPS to facilitate celestial body coordinate frame computations.

Table 4. CLVTOPS Models for Composing TREETOPS Derivatives.

Model	Description
Environment	Parameters (wind speed, speed of sound, etc.) for other models
Trajectory	Parameters (altitude, angle-of-attack, Mach #, etc.) for other models
Aerodynamics	Forces and moments
Mass Properties	mass, center of gravity (CG), moment-of-inertia for each body
Gravity	Gravitational forces for each body
Mission Manager	Polls and triggers vehicle and simulation events
Engines	Thrust, mass flow rates, transients
Guidance, Navigation, & Control (GN&C)	Actuator commands (thrust vector control (TVC), engine throttle, on/off, events)
Sensors	Sensor dynamics and error source models
Actuators	Thrust direction with error sources & dynamics
Custom Forces	SLS/ML specific forces such as umbilical separation, pad release, slosh dynamics
Output	Instrumenting for simulation variables

Table 3. CLVTOPS High-Fidelity Modeling Highlights.

Model Feature	Description
Gravity	Uses Pines' Method[31] for geopotential with (for SLS analysis) Earth-based spherical harmonics coefficients from the Gravity Recovery And Climate Experiment (GRACE) Gravity Model 02 (GGM02) [32], where the user can specify the degree and order depending on the desired fidelity. Gravity models for other celestial bodies and simpler fidelity gravity models are also available and configured through data inputs.
Time	Available start-time formats: Coordinated Universal Time, Barycentric Dynamical Time, Terrestrial Dynamical Time, and most common formats (calendar, day of year, and Julian Date). SPICE[30] keeps track of the time in the simulation, sets the correct true-of-date inertial frame to earth-centered-earth-fixed frame transformation, and tracks planetary ephemerides through time. An International Astronomical Union 2000A precession-nutation algorithm that predicts the difference between Universal Time and Coordinated Universal Time, using Earth Orientation Parameters is also included[33].
Aerodynamics	SLS booster separation aerodynamics[34], based on Computational Fluid Dynamics (CFD) and wind tunnel tests, and includes 8 dimensional tables for the aerodynamic, Core Stage Engine and BSM plume impingement forces and moments for the separating boosters and remaining vehicle. SLS liftoff aerodynamics[35], including lumped and distributed aerodynamics, based on CFD and wind tunnel tests, which also includes the effect of the launch tower as a function of wind direction and vehicle altitude.
Atmosphere	Several earth-based atmosphere and wind models, including: Patrick AFB 1963 atmosphere model[36] Earth Global Reference Atmospheric Model 2010 (Earth GRAM 2010)[37] Doppler Radar Wind Profiler[38] Measured Kennedy Space Center ground winds
Mass Properties	Tracks propellant location, propellant remaining, interfaces to Engines module that will disable thrust if propellant remaining is zero, for example.
Engines	Can select from user-defined and SLS custom engine input models, including SLS-specific high-fidelity models for the booster ignition and burnout phases (based on Space Shuttle history and development motor ground tests), critical input for the liftoff and booster separation analysis.
Actuators	Includes high-fidelity non-linear simplex mechanical feedback models of the TVC system that will be used for the SLS CS engines and SRBs.

A feature and important part of the SLS computations is execution of the Guidance, Navigation and Control (GN&C) flight software. The actual C++ flight software code is ported in to run in CLVTOPS at the appropriate update intervals. Primary outputs of the GN&C software are SRB nozzle deflection angle commands, core engine nozzle deflection and throttle commands, and event triggering commands (SRB separation, for instance). CLVTOPS has several high-fidelity models to react to and stimulate the GN&C software. High fidelity sensor models for the Redundant Inertial Navigation Unit (RINU) and Rate Gyro Assemblies (RGAs) interact with "truth" motion states from TREETOPS to provide input to GN&C. High-fidelity actuator models react to the GN&C commands to provide inputs to and send forces and moments back to TREETOPS.

CLVTOPS takes advantage of TREETOPS' generic multi-body flexible structures formulation to easily model vehicle structural dynamics (flex), propellant slosh dynamics, and the inertia effects due to engine gimballing (known

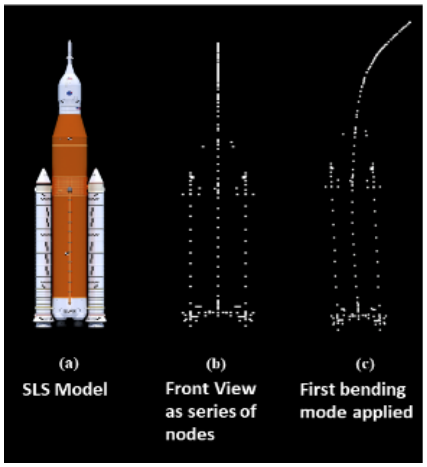


Figure 8. Integrated Vehicle Flex Example.
Bending is exaggerated for clarity.

as "tail-wags-dog/dog-wags-tail"), all without the user needing to change the TREETOPS equations of motion. Propellant slosh dynamics and engine inertia coupling are modeled by adding separate TREETOPS bodies for the slosh and engine masses. Modeling this phenomena in a non-multi-body simulation would require evaluation of the constraint and contact forces at each time step. The usage of Kane's Equations by TREEOPS allows the seamless integration of multiple connected piece parts while eliminating computational complexity by getting rid of the algebraic step of solving for these constraint forces.

Flex modeling principally addresses interaction with the closed-loop control system. Generally only mass-normalized orthogonal flex modes below 25 Hertz are considered since they, by far, have the most effect. Figure 8 shows one of the first flex modes for the integrated vehicle that are typically used for analysis. Four different structural dynamics models are used depending on the analysis case. Modes for the integrated launch vehicle are used for liftoff and separate modal models for the boosters (2nd and 3rd set) and CS (4th set) are used for stage separation analysis. The modes are frequency-sorted and down-selected based on their contribution to vehicle dynamics-related metrics. For example, a frequency-sorted down-select is performed on the first 300 modes of the integrated vehicle. This results in a nominal vehicle flex model representing 300 mass-normalized orthogonal modes (including slosh modes) up to 20.3 Hertz. The ML and vehicle models are connected prior to liftoff using an equivalent spring-based ML to model the effects of ten flexible ML contact points acting on the vehicle and four Vehicle Stabilizer System (VSS) radial retraction strut nodes. The equivalent spring-based ML model allows modeling of the combined ML and vehicle flex effects without using a large number of traditional modes. Also, the modeling of booster pressurization is included which couples with flex to influence the initial rise rate of the vehicle. Selection criteria for the flex mode set was partly chosen to ensure the effect of booster pressurization on vehicle flex was appropriately captured.

Multiple spring-mass-damper slosh models are used in concert with the flex modeling. These include the CS, Interim Cryogenic Propulsion Stage (ICPS), and Orion oxidizer and fuel propellant tanks. The slosh masses are modeled as separate TREETOPS bodies and are connected to the vehicle through hinges with only two lateral translational degrees of freedom. CLVTOPS computes and updates the slosh mass, spring constant, and damping coefficients based on the current vehicle states; such as axial acceleration, and TREETOPS propagates the lateral translation of the slosh masses. The effects of flex-slosh coupling are introduced by mapping the flex data from applicable slosh grids to the respective CLVTOPS flex node grids.

Finally, "tail-wags-dog/dog-wags-tail" (TWD/DWT) modeling is included for the four Core engine nozzles and the two SRB nozzles. The nozzles are modeled as separate TREETOPS bodies and are connected to the vehicle through hinges with only two rotational degrees of freedom, along the nozzle actuator rotation axes. For each nozzle, representative rotational spring stiffness and viscous damping values are assigned to the two rotational degrees of freedom. The stiffness and damping values are extracted from each nozzle's thrust vector control mechanical actuator model. Actuator torques are computed and extracted from each nozzle's actuator model and are applied directly to the hinge axes. TREETOPS propagates the nozzle rotational dynamics, and the nozzle attitude is fed back to the mechanical actuator model to use in the next integration cycle. Flex force-following coupling is achieved by interfacing the TWD/DWT model with the appropriate vehicle flex nodes.

CLVTOPS provides user-configurable output to support and instrument the model parameters-of-interest. A number of separate output streams can be configured, tailorable to a specific class of parameters or a specific category of output interpretation and display products.

This narrative is only a partial breakout of the comprehensive list of phenomena modeled in CLVTOPS. The following section, "Data Products", categorizes and lists parameters modeled in CLVTOPS that affect flight.

C. Runtime Scripting

This layer of tool chain addresses simulation execution and control. Separation of the model from its execution control is in concert with the well-known model-view-controller (MVC) software architecture[39]. This modern software engineering approach, now a de facto standard software construction practice, emphasizes separation of the

software tool (i.e. CLVTOPS/TREETOPS) from its control code. This approach recognizes, in the case of this SLS tool, that the practice and expertise of physics and algorithm modeling is distinctly different from the "nuts and bolts", more computer science-oriented domain of computer systems programming. Stated more simply, math modelers need only be concerned with coding the simulation itself leaving the computer execution mechanics to a separate domain of operating-system, systems experts.

Runtime scripting is accomplished with a versatile set of Python and Bash scripts that allow easy configuration and archiving of input data sets. A top-level schematic of the scripting system is shown in Figure 9. TREETOPS uses its native .int file that defines the system in terms of the tree topology's bodies and hinges. CLVTOPS uses a configuration (.cfg) file with data sets to define inputs for its models (described earlier). File indirection is used for both sets of inputs to promote reuse and organization of data, as well as to avoid redundancy of repeating the potentially very large data sets.

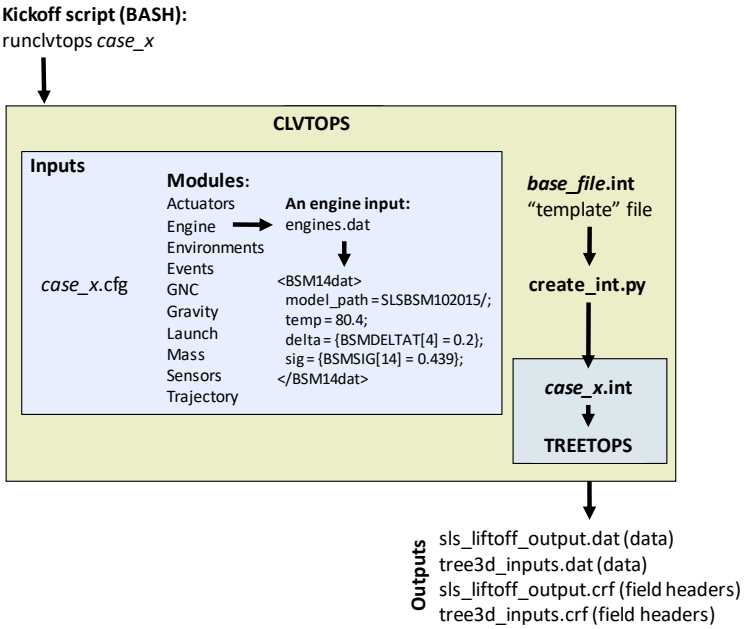


Figure 9. CLVTOPS Execution Schematic. *A mix of Python scripts, text data files, and Bash scripts are used to configure simulation inputs and organize output as calculations are completed.*

For TREETOPS, the data file is not directly configured but, instead, is created by a Python script that contains Python dictionaries of differences from a base file. Again, this system minimizes replication since the file indirection provides a mechanism to specify and conveniently catalog changes instead of having a myriad of input files where only a small set of parameters change.

A similar file indirection organization is used for the CLVTOPS input. A configuration file, in effect, provides a series of "pointers" to input files across the many model categories contained within CLVTOPS. Not shown in the figure is an additional file indirection with the capability to selectively change parameters in the input files. This capability is useful to create "dispersions" for multiple CLVTOPS executions to perform Monte Carlo and parametric analyses. The dispersions are created by Python scripts that vary the inputs from run to run, based on user instructions in the additional file.

Finally, at a top level, all the input file indirection, input file creation, and execution is orchestrated by a combination of Python and/or Bash scripts. A server-based computer cluster is available consisting of 200 Intel processor cores which provide parallel execution of 400 processes (each core can simultaneously run two simulations at once with their hyper-threading capability). Other compute configurations are also available for hosting runs and, except for system-specific interface code and command sets, CLVTOPS is not limited to any one compute configuration. A simple application programming interface (API) is provided by the scripts for the user to specify and control the simulation execution process and parameters. Again, this emphasizes the value of CLVTOPS MVC architecture, which enables an analyst to leverage the power of large-scale computation with only minimal systems-programming expertise.

D. Post-Processing

The post-processing is an important aspect of CLVTOPS owing to the number of and complexity of the diverse range of outputs and options for interpretation. Again in line with the MVC architecture, the post-processing is the "view" component, segregated from the other model and controller components. The post-processing "layer" is the most diverse, employing a variety of tools including MATLAB®[40], Python[41] and its associated Numerical Python, or NumPy, library[42]‡‡‡, open source plotting tools, including but not limited to gnuplot[43], Matplotlib[44], the

‡‡‡ Widely-used distributions, such as Anaconda [www.anaconda.com], are available for integrating these and similar tools and have been found to be extremely convenient so that all CLVTOPS analysts have a common toolset and implementation.

Visualization Toolkit (VTK)[45], and a Minimum Distance Tool to calculate launch vehicle separation distances and clearance. Scripting languages and MATLAB® are used extensively to collect, tabulate, and reformat CLVTOPS output.

The volume and variety of output from CLVTOPS presents a data management challenge and can quickly become untenable. Both Python and MATLAB® are well equipped to address large data processing tasks including transforming and mapping data from one "raw" data form into another format to make it more appropriate and valuable for a variety of downstream analytical purposes. These tools greatly simplify tabulation of statistics as well as structuring the data into serial objects that can be recovered from storage in their native structured format (Python pickle module[46], for instance). This data serialization capability is a powerful utility that greatly speeds up file input/output (speed-ups of greater than 10x are typically observed) and reduces file size compared to standard ASCII text. CLVTOPS also incorporates the ability to create compressed .hdf5 and .gz data files that facilitate post-processing[47].

The MATLAB® and open-source plotting tools expedite visual portrayal of the data with a wide variety of plotting representations normally used including x-y plots, histograms, and plot overlays superimposed on line drawings or illustrations of SLS hardware and components.

Tabulation of the complicated motions and geometries of the SLS liftoff and staging events requires efficiently computing clearances and determining the occurrence of recontacts between various hardware components (the tower, LPS, and boosters after separation, for instance). A Minimum Distance Tool, comprised of a set of Python scripts and a compiled C++ executable, combines the trajectory data (position and orientation) from CLVTOPS simulations with 3D mesh models derived from Computer-Aided Design (CAD) models, and determines the minimum separation distance between user-specified pairs of 3D mesh models through time. The tool uses rigid body trajectory data (position and orientation) and can also use flexible body motion (nodal displacements and rotations) to dictate the motion of 3D models. At the heart of the actual clearance computations is a general-purpose collision detection, distance computation, and tolerance verification library called Proximity Query Package[48], which essentially computes the minimum distance between sets of triangular mesh surfaces in three-dimensional space in a very efficient manner. The mesh surfaces originate from CAD models of the ground structures and SLS hardware that are pre-processed to produce an outer mold line – a water-tight shell model representing only the outermost surfaces of a CAD part or assembly. Actual clearances between the pairs of models, as well as the 3D points on each model between which the minimum clearance occurs, are sent to a data file.

Animated 3D visualizations of the separating bodies, including line segments representing time-varying minimum distances, are portrayed with TREE3D. This tool provides the ability to create engineering-based realistic animations similar to other commercial packages (like the Systems Tool Kit (STK)[49]). TREE3D is a Python tool developed with the VTK open-source software. Within TREE3D, Python scripts facilitate post-processing of CLVTOPS trajectory output for the visualization in concert with 3D mesh models derived from CAD models. The animated output from TREE3D is easily captured for incorporation into reports or briefing slides. Figure 10 highlights TREE3D's capabilities, showing a clip from a booster separation animation. Some impressive TREE3D fidelity is shown, including detailed Earth/atmosphere/lighting modeling and the ability to apply skins to the TREE3D objects (booster markings). Imagery created with TREE3D (like Figure 10) is also useful for determining and analyzing both ground-based and on-board camera placement, orientation, and field-of-view, which can aid in preparations for post-flight reconstruction.

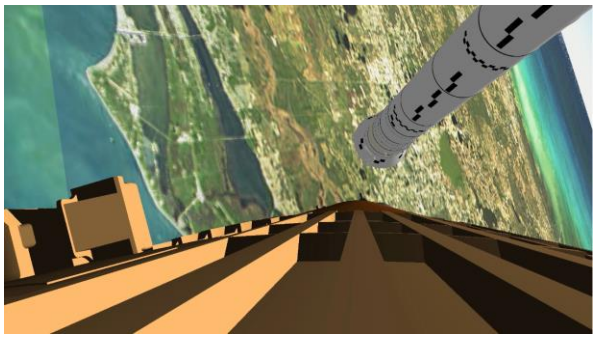


Figure 10. TREE3D Modeling Tool Output. *The TREE3D tool is capable of ingesting CLVTOPS trajectory data and generating animations and figures with a remarkable degree of fidelity.*

V. Data Products[§§§]

For the first SLS Block-1 flight, Artemis I, the principal data products from CLVTOPS fall into two categories: liftoff and booster separation. Ultimately, the analyses supported by CLVTOPS are used to demonstrate compliance with detailed verification objectives (DVOs) established by the program.

[§§§] All data products are representative and do not portray any SLS design qualities or performance.

Table 5. CLVTOPS 6DOF Simulation Error Sources.

Subsystem	General Category	Examples
Core Stage Engine and Booster Motor	Propulsion	Thrust, mass flow rate, pressure
Core Stage Engine and Booster Motor	Thrust vector	Misalignment (static & dynamic), application point
Core Stage Engine and Booster Motor	Thrust vector steering	Mechanical actuator servo parameters, compliance, damping
Booster Separation Motor	Propulsion	Thrust, mass flow rate
Booster Separation Motor	Thrust vector	Misalignment (static & dynamic), application point
Booster Separation Motor, Pyro Separation Hardware	Timing	Latency
RGA and RINU hardware and software	Navigation	Latency, location, misalignments, gyrocompassing, drift, noise, quantization, bias
Avionics hardware	GN&C	Computation latency, signal lags to actuators
Airframe	Aerodynamics	Force & moment uncertainty, plume impingement
Airframe	Mass properties	Mass, CG, moment-of-inertia (wet & dry)
Structure	Flex	Modal frequency and mode shape and slope uncertainties
Airframe Tanks	Slosh dynamics	Uncertainty for slosh frequency, mass, location, and damping
Flight Environment	Atmosphere & wind	Profile uncertainties
Mobile Launcher	Airframe Interaction	Compliance, applied force/moment uncertainties, disconnect timing
Airframe	Geometry tolerances	Variation due to thermal, stacking misalignment, form
Launch Planning	Day-of-launch	Timing variation for acceptable flight envelope and safety, open-loop guidance commands based on variation of design winds

The CLVTOPS standard analysis process is to assess the variability (or uncertainty) of all the model parameters and then conduct "dispersed" Monte-Carlo (MC) runs to generate statistics for the key technical performance measures. Generally, the dispersed parameters can be divided into two categories: 1) those associated with variations in the hardware and 2) subsystem performance and phenomenological uncertainties. Sometimes both of these categories are lumped into the term "error sources". For CLVTOPS, these equate to variation in the model inputs from run to run. These error sources can be biases, noise, probability of failure, or deviation-from-nominal, among others. Furthermore, these error sources can be one-time draws at the start of simulation or draws as the simulation runs (noise, for instance). Table 5 is a top-level listing for the SLS vehicle and flight error sources that are typically modeled in CLVTOPS. The quantity and detail of these error sources provide an indication of the level of fidelity of the CLVTOPS models.

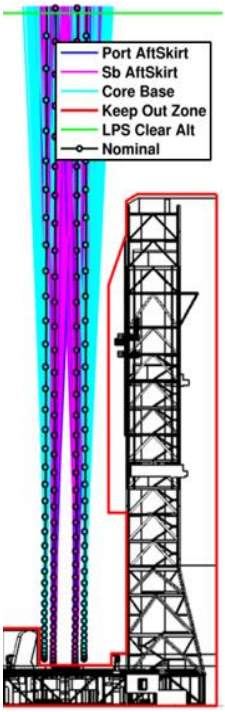


Figure 11. Liftoff Drift from CLVTOPS MC Runs. *Visual representation of clearance between elements of the SLS vehicle stack and the ML KOZ.*

One question that arises for these analyses is how many MC CLVTOPS runs constitute a confident assertion that requirements are met? For each of the liftoff and separation analyses, the primary baseline statistical parameter used to quantify the bound of the various important metrics is a program-designated percentile (>99%) with either 90 or 50 percent CL value, or 10 or 50 percent consumer risk (CR) value, respectively, regardless of distribution type or whether the data is one-sided or two-sided (note CL=100%-CR)[50]. Generally, 90 percent confidence is used for requirements compliance, such as clearance values, and 50 percent confidence is used for outputs that may be used as inputs to subsequent analyses, such as the vehicle states at separation. It was found that 2000 runs for each case provide enough samples to satisfy these criteria.

Following is a sampling of typical CLVTOPS analysis products that highlight CLVTOPS capabilities. While not exhaustive, the intention is to illustrate the variety and format of information CLVTOPS can generate. Most of the products selected for presentation here are pictorial in nature, but, underlying this, CLVTOPS generates extensive numerical data that can be mined to address specific analysis questions and domains.

A. Liftoff

The CLVTOPS analyses shown here demonstrate successful liftoff and tower clearance, and that the vehicle will not intrude into the static KOZ during liftoff.

Figure 11 is a visual representation of the primary CLVTOPS analysis product demonstrating successful liftoff and tower clearance and that the vehicle will not intrude

into the static KOZ during liftoff. Depicted is the drift of the SLS vehicle in the region of the launch tower in the North-Up plane. For each run, the path of the southern and northern critical points of the CS base, port SRB aft skirt, and starboard SRB aft skirt are plotted. The drift accounts for both the translation of the CG and the rotation of the vehicle. Note how the minimal tower clearance occurs near the top portion of the tower KOZ that contains retractable umbilical arms. Similar plots are generated for the failure conditions of CS engine out and nozzle actuators stuck at maximum deflection, among others.

Figure 12 is an overhead view of liftoff drift at the top of the LPS. Clearance to the LPS is most affected by wind, given the longer time to clear the higher-altitude structure.

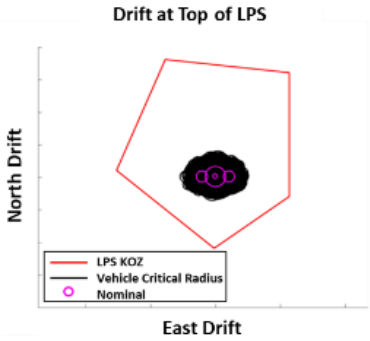


Figure 12. Liftoff Drift at Top of LPS. *Top down view of predicted vehicle drift in relation to the LPS KOZ.*

Nozzle drift just after liftoff is a key concern and is shown in Figure 13. The intention is to demonstrate clearance to the vehicle support posts depicted by a maximum volume outline. The dispersed trajectories are color-coded with respect to the port and starboard boosters. Plotting all 2000 trajectories, instead of only the outer worst-case bounds, provides additional visual cues about the distribution and division of dispersion between the two boosters.

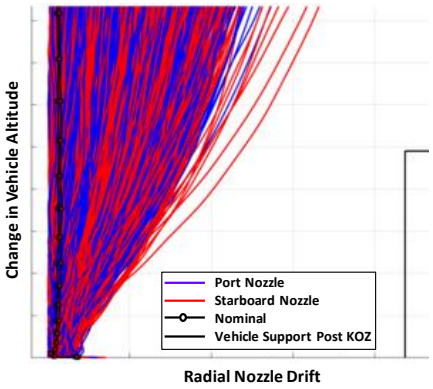


Figure 13. Nozzle Drift vs. Change in Altitude. *Visual representation of the maximum drift of both SRB nozzles toward the Vehicle Support Post KOZ.*

The analysis product in Figure 14 shows the minimum distance for 2000 cases between the 3D model of the vehicle and SRB nozzles, and the 3D model of the ground structure KOZ (Figure 3), as a function of time. The minimum distance associated with the base of the CS and each booster nozzle are computed for each trajectory using the Minimum Distance Tool described earlier. The minimum distance will decrease until the point of closest approach is achieved, then will increase as the vehicle moves further away from the object.

Focusing on the plume effects, Figure 15 depicts plume impingement projected to the ground until the vehicle clears the top of the tower. While not intended for clearance assessments, this data is available as an analysis input to other analysis teams for ML deck loads and acoustics.

While strictly not a requirements compliance product, the TREE3D tool described earlier is easily populated with the trajectory data for a 3D, solid-shape animation. Figure 16 is a screen-grab of an animation showing the vehicle's motion relative to the tower's KOZ. In this case, the animation was used to augment other CLVTOPS analysis for demonstrating the vehicle would clear the tower with a worst-case initial tilt.

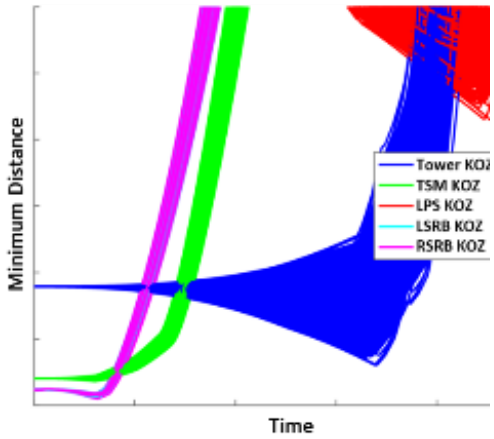


Figure 14. Minimum Distance to Ground Structures. *Color coded lines indicate calculated clearance between the SLS vehicle stack and various ground structures during the liftoff event.*

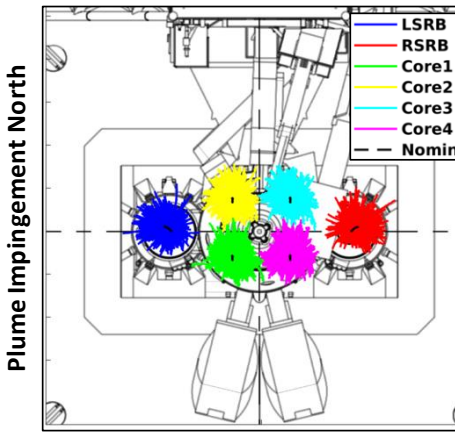


Figure 15. Plume Impingement Ground Track. *Colored areas indicate predicted nozzle plume coverage through tower clear.*

Another valuable analysis piece that can be "mined" from the CLVTOPS MC analyses is discovery of which dispersed parameters are the most significant. The scripting language post-processing tools described earlier (i.e. Python/NumPy and MATLAB®) are used to process the trajectory dispersions. The dispersions are sorted by relevance with respect to each measure of interest, e.g. vehicle drift, time-to-liftoff, etc. and correlations determined. Any correlation with a coefficient below 0.15 is typically omitted as there is a significant probability that such a correlation is noise-induced and not valid. A negative correlation coefficient denotes that the dispersion is negatively correlated with the output variable (e.g. SRB steady state chamber pressure is inversely correlated with time to clear tower, giving it a negative correlation coefficient). Table 6 is an excerpt from a correlations analysis to discover key "drivers" of the time-to-clear the tower. Time-to-clear the tower is an important metric since it is during this time window that collisions may occur.

Figure 16. TREE3D Output Example. *Representation of a liftoff trajectory with an initial vehicle misalignment.*

Table 6. Representative Correlation Analysis for Time to Tower Clearance.

Correlation Value	Dispersion Name
-0.8	Booster Steady State Chamber Pressure
0.2	Booster Ignition Delay

The standard analysis suite of plotted data is easily produced, including time histories, bar charts, scatter charts, and histograms, among many others. The CLVTOPS tool chain contains a suite of tools for this purpose with many choices. Or, the data can be easily aggregated and formatted to any other analyst's tool of choice. One example is the histogram shown in Figure 17 depicting the variation in time-to-clear the tower.

An additional CLVTOPS role is to provide valuable information for other disciplines to analyze individual subsystem performance. Figure 18 is an example MC dispersion for the SRBs' thrust vector control angles. This kind of data provides independent confirmation to the GN&C designers that performance is within as-designed nominal limits. In addition, the data confirms correct implementation of the GN&C flight software algorithms into CLVTOPS (with successful matching of GN&C team predictions).

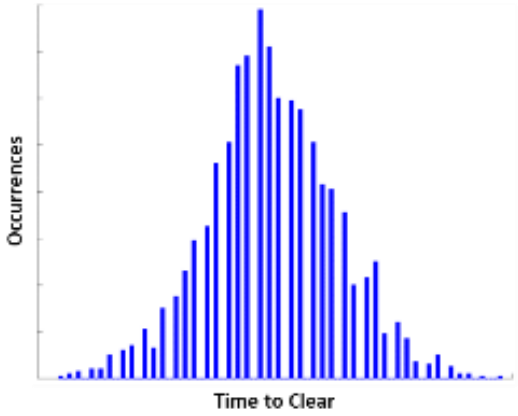


Figure 18. Histogram of Time to Tower Clearance.

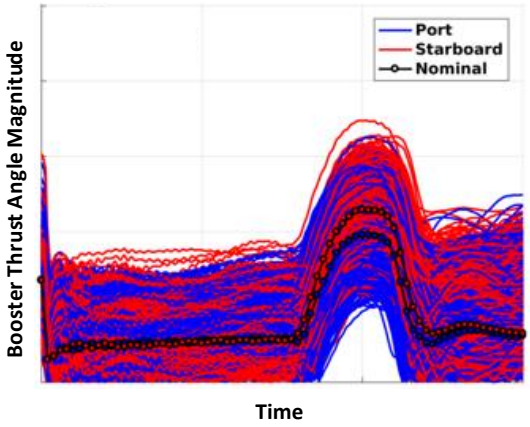


Figure 17. Booster Thrust Angle Magnitude for 2000 MC Dispersed Runs.

Beyond the brief listing of CLVTOPS visuals and data shown here, quantifying the effect of key model inputs on clearance and generating partial derivatives of those inputs with respect to clearance is a valuable tool for guiding CLVTOPS studies as well as guiding decision makers. The CLVTOPS post-processing tools can be applied to produce these "partials" of how varying one input can affect a critical output parameter. This is analogous to populating a Jacobian matrix where each partial is an element. Figure 19 illustrates one example showing tower clearance as a function of wind speed, which is conveniently linear (not always the case for other sensitivities). Therefore, a sensitivity partial could be devised for tower clearance reduction as a function of wind speed increase. The value of these partials is to guide analysis and not be a substitute for comprehensive CLVTOPS (or other analytical) investigation.

Monte-Carlo liftoff clearance analyses with respect to wind speed and direction are conducted to develop wind placards to support launch commit criteria. Non-failure and failure conditions are included in the analyses to develop a wind placard that results in adequate clearance margin, without sacrificing significant launch availability.

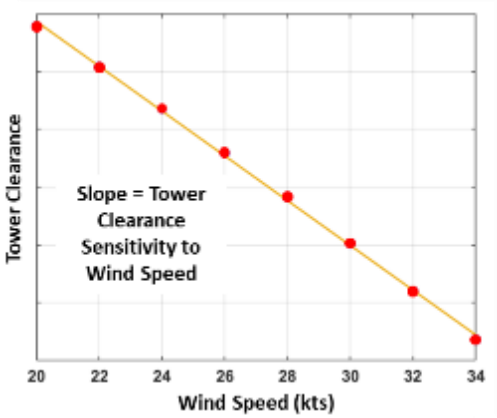


Figure 19. Tower Clearance Sensitivity to Wind Speed and Correlation Expression.

B. Booster Separation

The second category of critical CLVTOPS investigation is acceptable separation of the two side-mounted SRBs. As with the preceding narrative, the following is a sampling of typical CLVTOPS analysis products. Again, the intention is to illustrate, non-exhaustively, the variety of CLVTOPS tool chain products that address launch system requirements compliance. Since methodology is already addressed in the liftoff discussion, only the booster separation analysis products are described here.

The booster separation sequence is triggered when threshold pressures are reached in the SRBs as they burn out, roughly two minutes into flight. Four small BSMs each on the fore- and aft-ends facilitate separating from the CS. Booster separation dynamics are simulated in CLVTOPS as three bodies, held together by high-stiffness/damping devices that are cut at separation time. By immediately changing the stiffness to zero at separation, the two SRBs become independent bodies, and clearances between these three bodies are computed using the Minimum Distance Tool with 3D mesh models of SLS hardware. Figure 20 is a screen-grab from a TREE3D animation, generated with CLVTOPS data, of the boosters separating. Clearance distances between critical components and locations are shown, using data generated from the Minimum Distance Tool.

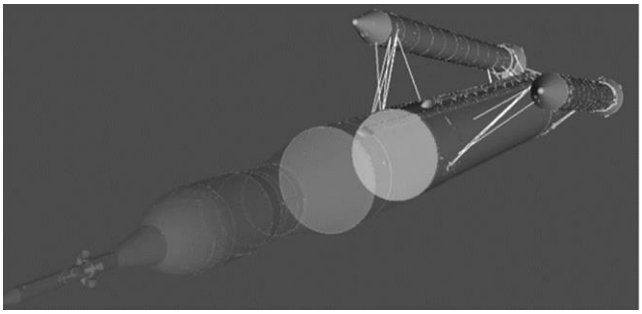


Figure 20. Booster Separation Animation. *White lines indicate calculated distance between various points of CS and SRB geometry.*

Timing requirements for the aforementioned booster separation sequence are determined with CLVTOPS analysis and are used directly by the flight software. One key requirement is a pre-determined separation (time) delay between when a threshold pressure is reached within either SRB motor and when the actual separation command is issued. This delay allows SRB thrust to "tail-off", so that when separation occurs, the SRBs will quickly fall away from the CS, and also allows the CS to hold attitude throughout the separation event. This time delay impacts the clearance between each SRB and the CS; a shorter delay might decrease clearance, therefore increasing the chance of recontact, but would allow more payload to be carried, as the SRBs would be released a few seconds sooner. A longer delay might increase clearance, but would come at the cost of payload performance (the CS would be essentially "dragging" the spent SRBs for a longer period of time, burning propellant needed to carry payload mass to orbit). The final values chosen for this time delay are selected through an iterative process using CLVTOPS, balancing the risks of CS/SRB recontact with desired payload performance.

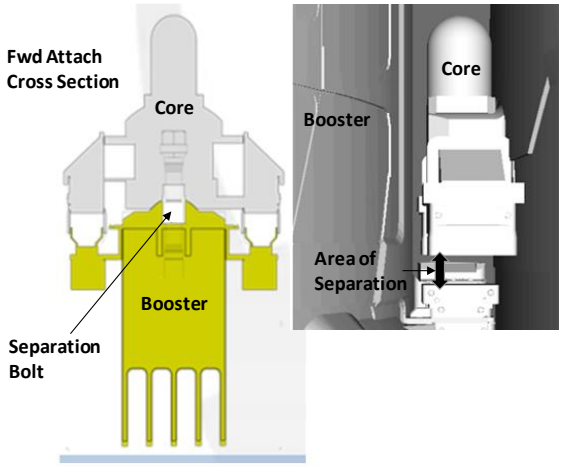


Figure 21. Forward Ball-Socket Assembly. *Clearance between the CS and SRB forward attach is one of the tightest areas addressed in SLS analysis.*

To assess these risks, three technical performance measures (TPMs) are used as standards for assessing the amount of margin with respect to the booster separation clearance requirements. Each booster is attached to the CS at two locations: a ball-socket joint at the forward end (first

TPM) and three struts at the aft end (second TPM). These are illustrated in Figures 21 and 22, respectively. The initial clearances as these two attachments separate are very tight, on the order of fractions of an inch. The macro clearance (third TPM) is shown in Figure 23. The boosters must clear the aft end of the CS with a minimum clearance margin.

CLVTOPS accurately models the booster motions after separation. Figure 24 is a screen-grab from a TREE3D animation showing one typical case. As with liftoff analysis, 2000 trajectories with dispersed error sources are used to generate a set of samples. A major analysis product is shown in Figure 25 that graphically depicts an aft strut clearance time history after the boosters separate. None of the trajectories over time intersect the bottom of the y-axis scale (i.e. 0), indicating no collisions. As intuition suggests, all of the curves increase with time as the booster falls away. Again, the color coding for the boosters and CS are useful visual cues for their distribution within the three-sigma bounds. Figure 26 is a similar plot for the forward ball-socket attachment. For both plots, the Minimum Distance Tool took care of the large amount of computations required to sample clearances between all the points of interest to find the smallest one (at each time step!).

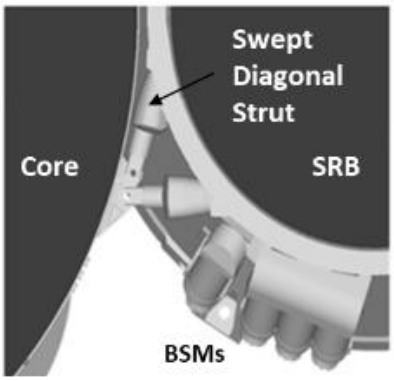


Figure 22. Aft Strut Assembly. *Two struts are visible; the "diagonal" (middle) strut, and the "upper" strut. For clearance analyses, strut ends are modeled as swept volumes, created by articulating the strut geometry through its entire possible range of motion.*

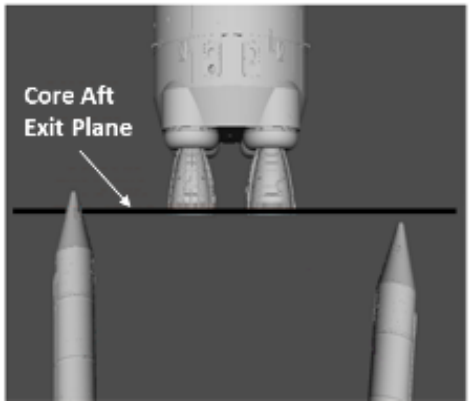


Figure 27. Core-Booster Macro Clearance. *The minimum distances between the CS and each SRB when each SRB nose cone passes through the CS engine nozzle exit plane indicates more large-scale, or far-field, clearance.*

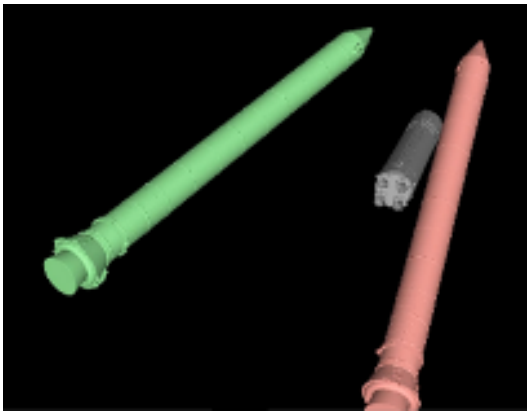


Figure 28. Booster Trajectory after Separation.

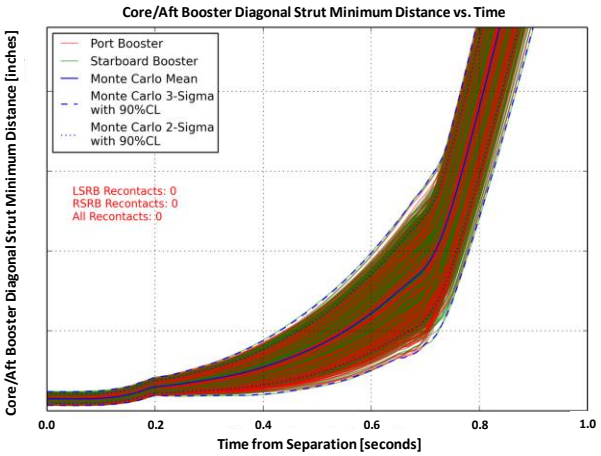


Figure 25. Core Aft/Booster Diagonal Strut Minimum Distance. *Positive values indicate there is no recontact after detachment of the diagonal strut.*

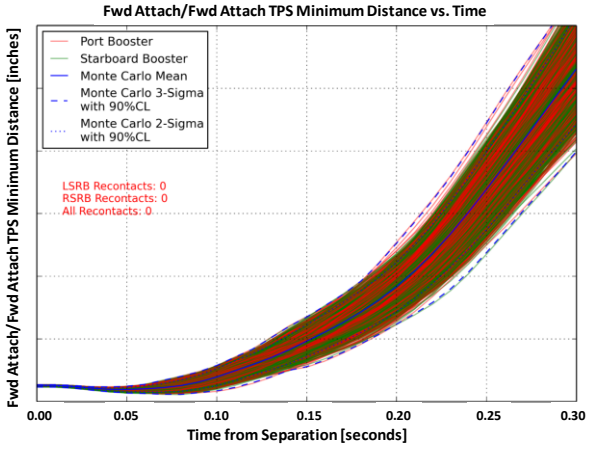


Figure 26. Forward Ball-Socket Clearance. *Minimum distance between SRB and CS Thermal Protection System (TPS) (foam/cork) at the forward attach ball-socket.*

The data portrayed in a plot, such as the preceding, is usually processed in a variety of ways by the CLVTOPS post-processing tool suite to generate specific statistics and aid interpretation efforts. For instance, the histogram in Figure 27 is an example of conventional statistics plots used to address booster separation.

TPM results are also typically studied and generated for failure scenarios. BSM failure and CS engine failure are used to generate similar plots as those previously shown.

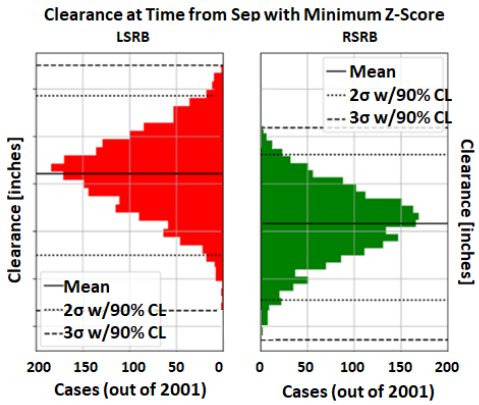


Figure 29. Minimum Booster Clearances. *Co-plotted histograms are a good way to compare output from both the left and right SRBs simultaneously.*

VI. Important Support Activities

A comprehensive support structure has been put in place to provide for reliable CLVTOPS verification, archiving, version integrity, and training.

A. Verification

Verification and validation of CLVTOPS and its constituent models is a continuous process to ensure accurate and credible results. This is ensured by getting data from credible sources, verification with independent simulations, and comparison to flight results, where possible.

All CLVTOPS model parameters are performance- or requirements-based and are obtained from accredited sources and activities, whose sources are directly linked to program documentation. All data transactions to and from CLVTOPS are documented.

The CLVTOPS verification activity is the process of confirming that it is correctly implemented with respect to the conceptual math models, matching agreed-upon specifications and assumptions. During SLS verification, its models are constantly tested to ensure commonality with other simulations and to find and fix errors. This activity is augmented by comprehensive team peer-review. VanZwieten[51] describes a methodology for a comprehensive time and frequency comparison between several SLS simulations including Marshall Aerospace Vehicle Representation in C (MAVERIC)[52] and Program to Optimize Simulated Trajectories 2 (POST2)[53], along with several others. MAVERIC is MSFC's prime 6DOF simulation for design, analysis, and performance prediction. It is also used by MSFC's GN&C team as a design platform for flight software design. POST2 is NASA Engineering and Safety Center (NESC) NASA-Langley's 6DOF for end-to-end simulation of launch vehicle trajectories. The NESC performs a separate independent verification and validation (IV&V) activity for SLS liftoff and booster separation analyses. Periodic simulation "shoot-outs" are conducted to demonstrate acceptable agreement between these models, providing confidence in CLVTOPS nominal model implementation. Single-run trajectories and dispersed results are compared where physical and phenomenological parameters are varied. All simulations must agree within specified thresholds for DVO compliance.

An example verification product is shown in Figure 28. This envelope plot is one of many comparing MAVERIC simulation statistical results with CLVTOPS (in this plot, dynamic pressure). Shown are 2000 trajectories from each simulation with one, two, and three sigma variations from each simulation. The envelope plot makes it easy to see that there is good agreement between the two simulations and their dispersion models.

Envelope plots exemplify a particularly useful but difficult-to-construct visual portrayal due to the large amount of data processing and data wrangling required. Here, several trajectories are "sliced" along the x-axis to tabulate consolidated statistics across many y-values. The slicing process can be greatly complicated when the independent x-axis values for the plotted parameters do not coincide, making it difficult to compute statistical parameters at any one x-axis value. Python's NumPy package and its associated array processing functions make this complex task easy to perform with minimal code.

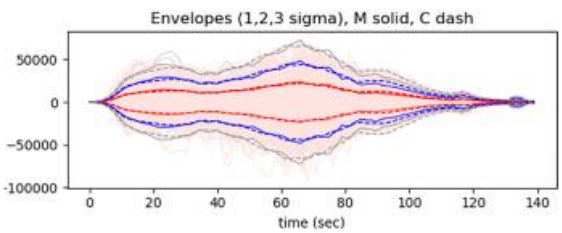


Figure 30. Comparison of CLVTOPS and MAVERIC with Envelope Plot.

As of the writing of this paper, SLS has never flown so no direct validation with flight data is possible.

However, ground testing ranging from small individual components all the way up to static firing of the engines provides data for a large degree of validation.

As referenced earlier, launch of the Ares I-X Flight Test Vehicle on October 28, 2009[12] presented a great opportunity to provide validation evidence for CLVTOPS using actual flight data. In order to simulate the Ares I-X flight, specific models were implemented into CLVTOPS. These models included the flight day environment, reconstructed thrust, reconstructed mass properties, aerodynamics, and the Ares I-X GN&C models. The resulting CLVTOPS output was successfully compared to Ares I-X flight data[12]. Figure 29 is an example data product from this activity.

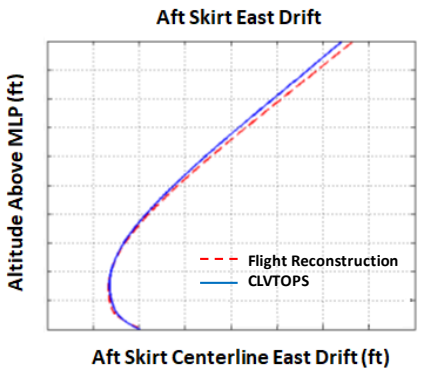


Figure 31. Aft Skirt Drift Comparison. *Output from CLVTOPS closely matched data reconstructed from the Ares I-X flight.*

B. Archiving

It is critically important to be able to go back and replicate previous CLVTOPS analyses, start from a previous version, or answer questions about precisely what configuration and data were used for previously reported analyses. The open-source Subversion (SVN)[54] configuration control system provides this critical function for CLVTOPS, its constituent tools, and data. Using SVN, every change made to the repository is saved, along with who changed it, the date, log message, and other information. The SVN repository makes it easy to revert back to earlier versions and is a record of changes and their details. A "trunk" and "branch" architecture facilitate spawning new versions and optionally integrating them back into the main analysis tool. All CLVTOPS changes are peer-reviewed by the CLVTOPS team before submitting.

C. Automated Version History

A key question in any collaborative development effort is ensuring that changes do not have any unintended side effects or produce questionable results. This complex activity can become tedious with large-scale programs like CLVTOPS and comprehensive testing can start to be neglected or not fully performed. CLVTOPS has automated the change and version monitoring process with its AutoMC tool. A Python script runs selected CLVTOPS MC cases and keeps a running list of tabulated statistics for the latest version. An automated report of key parameters is generated for the latest simulation configuration and compared against the previous report. This activity generally auto-runs "off-hours" to avoid interference with mainstream CLVTOPS workflow processes and runs, and an auto-generated e-mail with the results is sent to team members each week for review. This weekly review process greatly increases the probability that unintended consequences of submitted updates will be caught and remedied swiftly.

D. Training

Continued development and effective application of CLVTOPS requires training materials so that new users and developers can be added to the team in the course of normal staffing turnaround. A suite of training materials is available so that new team members can, largely with independent study, become acquainted with CLVTOPS. A cornerstone of the training is a Saturn V-like, two-stage-to-orbit example, where a CLVTOPS analysis is worked completely through, from configuration of input, to generating output, and finally interpretation of results. Several of the CLVTOPS tools are introduced and used including the Minimum Distance Tool, TREE3D, and scripting tools to post-process results. An additional collection of training briefings, some as Microsoft (MS) PowerPoint slides, describe step-by-step mechanics of using parts of CLVTOPS, such as the Subversion archiving described earlier. A "CLVTOPS Quick Start Guide" or "cheat sheet" similar to those found on the internet for many computer software tools can be consulted after learning CLVTOPS as a refresher or reference.

VII. CLVTOPS Broader Application

Although this paper focused on SLS Block-1 liftoff and booster separation, the CLVTOPS tool chain can be effectively applied to address many other aerospace multi-body problems. As elaborated on in this paper, the modular, MVC architecture of CLVTOPS allows components of its tool chain to be utilized and configured on a selective basis so that a CLVTOPS tool solution can be efficiently tailored to address a broad range of analyses. This section briefly describes some additional application examples starting with a well-known satellite spinning example, then progresses

to examples that address smaller SLS-related separation events, and concludes with a completely different space vehicle application.

A. Spinning Satellite

This example is shown to demonstrate how CLVTOPS is easily applied to smaller-scale, simpler problems without any modification of the tool chain structure.

Explorer 1 was a milestone in space flight history, being the first United States satellite. It is also notable that an unanticipated dynamics behavior was observed. Explorer 1 changed rotation axis after it was placed in orbit. The elongated body of the spacecraft had been designed to spin about its long (least-inertia) axis but refused to do so, and instead started precessing due to energy dissipation from flexible structural elements. Later it was understood that on general grounds, the body ends up in the spin state that minimizes the kinetic rotational energy for a fixed angular momentum (this being the maximal-inertia axis).

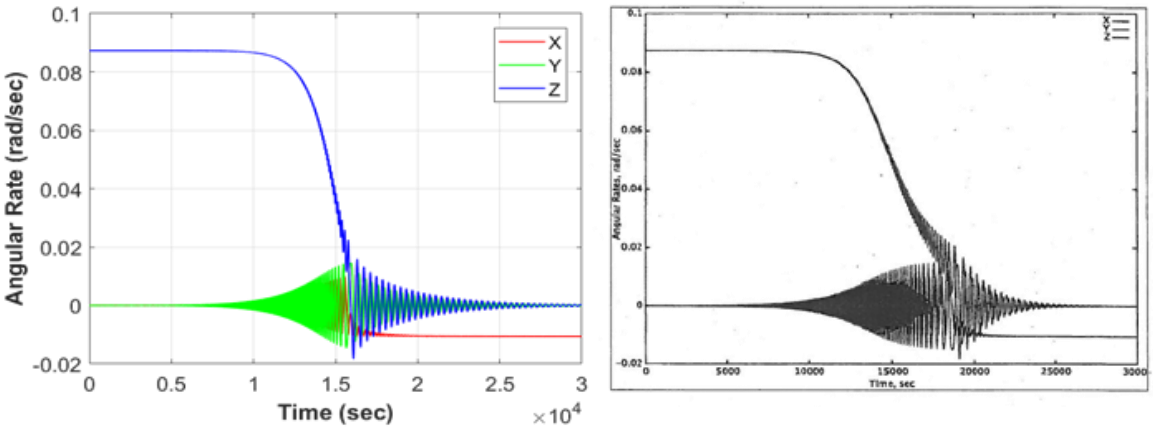


Figure 32. Angular Velocity Comparison. *(a) CLVTOPS; (b) Stoneking.*

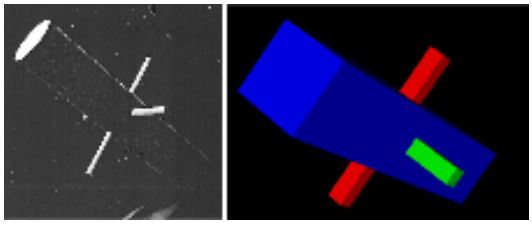


Figure 33. 3D Model from Stoneking and CLVTOPS of Satellite Spin. *(a) Stoneking; (b) CLVTOPS. (Note: Screen-capture times not exactly at same time in simulation).*

The TREETOPS component of CLVTOPS was used to replicate this behavior, repeating an example analysis by Stoneking[55], where Kane's method was also applied. Figure 30 is a side-by-side comparison of the angular velocity dynamics as the satellite progresses to a flat spin. Figure 31 is a side-by-side 3D model comparison. The CLVTOPS picture is a screen-grab from a TREE3D animation.

This application example also provides a good verification case for TREETOPS since the case is well-documented and has relatively complex dynamics involved.

B. SLS Payload Fairing Separation

CLVTOPS application to the SLS payload fairing separation highlights the in-depth dynamics analysis capability. Due to the highly flexible nature of large payload fairings, flex deflections need to be considered when examining the clearance between the separating fairing and the payload and launch vehicle. A generic approach was developed using Python to automatically segment a large 3D mesh model into multiple smaller 3D models that can each be translated and rotated according to the CLVTOPS outputs of the deflections and rotations from the closest flexible body node. Individual clearances between the payload and each of the smaller 3D models are calculated using the Minimum Distance Tool. The composite result is equivalent to having the original 3D mesh clearance model deform through time due to flexible body dynamics. Figure 32 shows two screen-grabs from TREE3D. Although it appears that the fairing halves are single 3D models, they are actually comprised of over a hundred smaller models, each anchored to individual flex nodes. The two screen-grabs were taken at different times from the same simulation run and highlight the fairing flexible body "breathing" or "pinching" mode response to the separation impulse. In the image on the left, the bottom edge corners can be seen "pinched in" compared to the image on the right, which was taken from a later

time. This "pinching in" effect is why the flexible body dynamics must be considered when calculating the clearances between the fairing and payload.

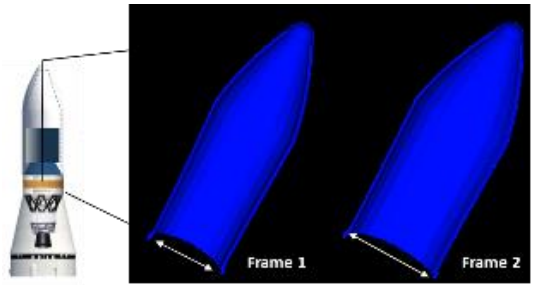


Figure 34. SLS Payload Fairing Separation. *Frame 1 shows the "pinched" mode response to the separation impulse; Frame 2 was taken later in the simulation and shows the non-trivial effect of flex dynamics on the payload fairing.*

C. Secondary Payload Dispensing

Another example shows CLVTOPS SLS application at a smaller scale than the full-up launch vehicle. In addition to carrying Orion as the primary payload on Artemis I, 13 six-unit cube satellites will be stowed and then jettisoned after the Orion spacecraft has separated from the SLS. These secondary payloads will be stowed inside spring loaded dispensers mounted inside of the Orion Stage Adapter. At the time of jettison, the dispensers will eject and propel the secondary payloads, which must fly through and clear the interior of the Orion Stage Adapter and Orion Spacecraft Adapter before beginning their individual missions. CLVTOPS was used to model the secondary payload jettison event to determine the payload clearance sensitivity to the vehicle attitude rates and to the spring dispenser jettison forces. The CLVTOPS multi-body dynamics feature was used to model the payloads and the vehicle as separate bodies and to track their relative motions during the jettisons. Figure 33 is a TREE3D animation screen-grab from CLVTOPS-generated data of four of the payload jettison events. The payload envelopes shown include potential appendages (such as solar panels) deployed. The Orion Spacecraft Adapter is shown in gray. The animations provide visual cues, to augment x-y plots, for the occurrence of contact as the vehicle attitude rates and spring dispenser forces were parametrically varied. Notably, the Minimum Distance Tool was used to calculate the minimum clearances between the payloads and the Orion Spacecraft Adapter as they evolved over time. These clearances are indicated by the colored lines in the figure.

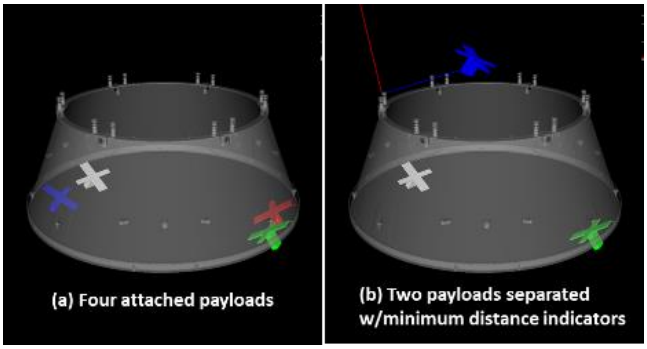


Figure 35. Cube Satellite Payload Separation Analysis.

D. Separation System Requirements

CLVTOPS is also used to help develop requirements for several new separation systems to be used in the SLS Block-1B configuration. It should be noted that the Block-1 liftoff and booster separation systems are primarily based on existing Shuttle heritage design, so did not necessitate new separation system design requirements. The Block-1B separation system requirements developed using CLVTOPS analysis include the CS/EUS separation system, the EUS/USA separation system, and the EUS/co-manifested payload (CPL) and primary payload separation systems. Figure 34 shows a screen grab from a TREE3D animation based on CLVTOPS outputs, which depicts a docked Orion (cone to the left of the red component) with a generic CPL (red component) separating together from the EUS (green and gray components on the right). These separation system requirements are usually generated from analyzing statistics from CLVTOPS MC dynamics and clearance results, where inputs such as the vehicle mass properties, vehicle attitude rates at separation, and generic separation system impulses and disturbances are all dispersed. CLVTOPS MC analysis can also be used to evaluate clearance performance for requirements that are generated by other means, such as dynamic constraints levied on the vehicle. The requirements typically specify the needed separation system impulse

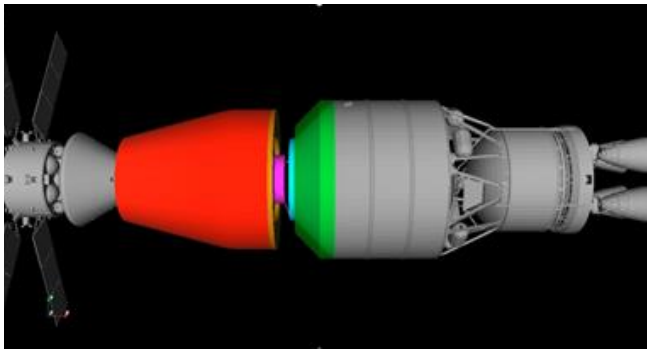


Figure 36. EUS/CPL Separation.

or energy while also specifying limits on the separation system tip-off torques or misalignments, in order to guarantee separation clearance while also meeting other dynamic constraints.

E. Europa Lander De-Orbit Stage Separation

The tool chain is extensible to celestial bodies other than Earth and is easily integrated with other 6-DOFs to produce a final product. CLVTOPS was used to get a full multi-body 6-DOF of the Europa Lander De-Orbit Stage (DOS)/Powered Decent Vehicle (PDV) separation event up and running very quickly, providing program managers and chief engineers credible models very early in the design process to support the Mission Concept Review. The DOS is a solid rocket motor used to de-orbit the lander from its trajectory over Europa, one of the four Galilean moons orbiting Jupiter. After the solid rocket motor is spent, the DOS is jettisoned and follows a ballistic trajectory until impact on the Europa surface. Figure 35 shows a TREE3D screen-grab of the separation event. The separation analysis involved examining the near-field clearances between the DOS and PDV (to ensure the DOS residual tail-off thrust does not force the stage to immediately recontact the lander) and the far-field clearances from the uncontrolled falling DOS and the maneuvering/controlled landing of the PDV. The analysis was used in the design of the separation scheme, including the number, placement, and orientation of the separation motors. The analysis presented several unique modeling challenges, including simulating a 6-DOF on a different celestial body other than Earth and integrating with another 6-DOF that modeled the combined DOS and PDV prior to separation but only the PDV after separation.

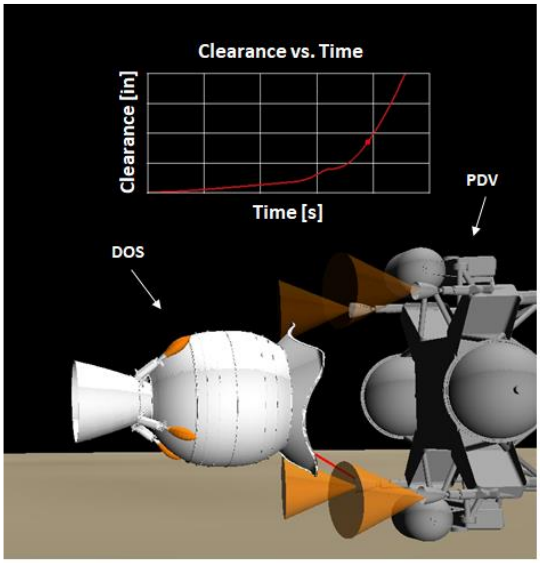


Figure 37. Europa Lander Separation. *TREE3D animation of the DOS separating from the PDV, including DOS separation motor placement and PDV plume interaction volumes.*

VIII. Summary

Software and simulation technology continues to advance in terms of ease-of-construction, flexibility, maintainability, and reuse and it is important for dynamics analysis tools to take advantage of these advancements. CLVTOPS supports and augments the SLS design and flight preparation process by advancing the simulation state-of-the-art with a flexible tool chain, anchored by established physics codes and libraries, that greatly reduces the cycle-of-analysis time - from question-asked, to analysis, to results that can be fed back into the next analysis and design cycle. The variety of post-processing products to mine the analysis data, along with the speed-to-completion, enables more iterations of the analysis cycle to more fully understand SLS performance, provide more information for decisions, and more fully leverage simulation's potential to demonstrate readiness for flight.

But perhaps the main contribution of this paper is to document and demonstrate a pattern for dynamics tool construction that can be applied to other aerospace simulations – namely, a model-view-controller architectural approach for decomposing and separating simulation parts into the physics and model engine (the *model*), provisions for a suite of tools that exclusively address data mining and presentation (the *view*), and separate software to configure input and control execution (the *control*). Traditionally, the gap between application of modern software engineering techniques and "ad-hoc" programs written and designed by engineers and scientists has been difficult to bridge. It is hoped that the design and implementation of CLVTOPS can shrink this gap by being an example of how large-scale simulation systems can have a solid architectural foundation and yet be accessible to analysts to modify, use, and tailor to their individual workflow processes.

While the scope of this paper is at a "survey" level, it sets the stage and provides a reference for future more in-depth documentation expanding on the details of the tools, techniques, and example data products described herein.

Acknowledgments

A simulation tool of this scope is necessarily the product of the contributions and collaboration of many talented individuals. The authors, current members of the Liftoff and Separation Dynamics team, would like to recognize the invaluable contributions of the CLVTOPS developers that preceded them. Dr. Young K. Kim created the initial version

of CLVTOPS during NASA's Constellation Program and led the team and CLVTOPS development during the Constellation Program and into the first four years of SLS development. Dane J. Childers led the team after Dr. Kim until 2019, and provided multiple invaluable contributions to CLVTOPS, including the development of TREE3D and incorporation of the Minimum Distance Tool into the tool chain. Other invaluable contributors include Victor M. Collazo-Perez, Jason M. Everett, Kevin W. Geohagan, John R. Glaese, Levin M. Guillermo, Joseph A. Huwaldt, Jon David Jackson, Robert J. Jurenko, Kyle A. Miller, Juan I. Orphee, John A. Ottander, Kristina D. Schwarz, Brandon C. Stiltner, Timothy D. Stuit, and John H. Wall. The authors would also like to acknowledge and thank the SLS System Analysis and Integration lead Joel (Joey) M. Broome, the SLS Vehicle Management leads Timothy Jason Bush, Robert A. Hall, and Dr. Robin M. Pinson, the MSFC EV40 Division Chiefs Anthony T. Lyons and B. Glenn Overbey, the MSFC EV42 Branch Chief Heather M. Koehler, the MSFC EV42 Flight Dynamics Team Lead Paul Von der Porten, and the entire MFSC EV41 and EV42 teams.

List of Acronyms

API	Application Programming Interface
BSM	Booster Separation Motor
CAD	Computer-Aided Design
CFD	Computational Fluid Dynamics
CG	Center of Gravity
CL	Confidence Level
CR	Consumer Risk
CS	Core Stage
DOS	De-Orbit Stage
DVO	Detailed Verification Objective
ESM	Encapsulated Service Module
ET	External Tank
EUS	Exploration Upper Stage
GN&C	Guidance Navigation & Control
GGM02	Gravity Recovery And Climate Experiment Gravity Model 02
GRACE	Gravity Recovery And Climate Experiment
ICPS	Interim Cryogenic Propulsion Stage
IMU	Inertial Measurement Unit
JSEG	Jacobs Space Exploration Group
KOZ	Keep Out Zone
LAPACK/BLAS	Linear Algebra PACKage/Basic Linear Algebra Subprograms
LPS	Lightning Protection System
MAVERIC	Marshall Aerospace Vehicle Representation in C
MC	Monte-Carlo
ML	Mobile Launcher
MPCV	Multi-Purpose Crew Vehicle
MSFC	Marshall Space Flight Center
MVC	Model View Controller
NASA	National Aeronautics and Space Administration
NESC	NASA Engineering and Safety Center
OML	Outer Mold Line
PDV	Powered Descent Vehicle
PLF	Payload Fairing
RGA	Rate Gyro Assembly
RINU	Redundant Inertial Navigation Unit
SLS	Space Launch System
SPICE	Spacecraft, Planet, Instrument, Orientation ("C-Matrix"), and Events information system
SRB	Solid Rocket Booster
STK	Systems Tool Kit
STS	Space Transport System
SVDS	Space Vehicle Dynamics Simulation
SVN	Apache Subversion

TPM	Technical Performance Measure
TPS	Thermal Protection System
TSM	Tail Service Mast
TVC	Thrust Vector Control
TWD/DWT	Tail Wags Dog/Dog Wags Tail
USA	Universal Stage Adaptor
USM	Ullage Settling Motor
VSP	Vehicle Support Post
VSS	Vehicle Stabilizer System
VTK	Visualization ToolKit

References

[1] "Appendix D - Saturn Launch Summary", URL: https://history.nasa.gov/MHR-5/app_d.htm [cited 30 April 2020]

[2] "NASA's Plan for Sustained Lunar Exploration and Development", URL: https://www.nasa.gov/sites/default/files/atoms/files/a_sustained_lunar_presence_nspc_report4220final.pdf [cited 30 April 2020]

[3] Foust, J., "NASA report outlines vision for long-term human lunar exploration", URL: https://www.nasa.gov/sites/default/files/atoms/files/a_sustained_lunar_presence_nspc_report4220final.pdf [cited 30 April 2020]

[4] Day, D., "Saturn's fury: effects of a Saturn 5 launch pad explosion", URL: https://www.thespacereview.com/article/591/1 [cited 30 April 2020]

[5] "Digital Program BHA0030-D, Saturn V/S-1C Flight Dynamics", The Boeing Company, Dynamic Analysis Group, 1967, revised 1969, building 4600, NASA Marshall Space Flight Center.

[6] Wasko, R. A., "Experimental investigation of stage separation aerodynamics", NASA-TND-868, 1961.

[7] Chubb, W., "The collision boundary between the two separating stages of the SA-4 Saturn Vehicle", NASA-TND-598, 1961.

[8] Decker, J., Pierpont, P., "Aerodynamic separation characteristics of conceptual parallel-staged reusable launch vehicle at Mach 3 to 6", NASA-TMX-1051, 1965.

[9] Decker, J., Gera, J., "An exploratory study of parallel-stage separation of reusable launch vehicles", NASA-TND-4765, 1968.

[10] "Space Shuttle Liftoff Separation Data Book, Revision A," Rockwell International, SSD96D0194, 1996.

[11] "Space Shuttle System SRB Separation Verification for Operations", Rockwell International, STS 82-0570, NASA contract NAS9-14000, 1982.

[12] Burger, B. S., Schwarz, K. D., and Kim, Y. K., "CLVTOPS Liftoff and Separation Analysis Validation Using Ares I-X Flight Data", JANNAF Paper 1838, 58th JANNAF Propulsion Meeting, April 2011.

[13] NASA Technology Transfer Program, URL: https://software.nasa.gov/software/MFS-33566-1 [cited 8 May 2020]

[14] Kane, T. R., and Wang, C. F., "On the Derivation of Equations of Motion", *Journal of the Society for Industrial and Applied Mathematics*, Vol. 13, No. 2, June, 1965, pp. 487-492, Society for Industrial and Applied Mathematics. URL: http://www.jstor.org/stable/2946443.

[15] Lanczos, C., "The Variational Principles of Mechanics", Courier Corporation, 2012.

[16] Vandervoort, R. J., "Modeling and Analysis of the Pinhole Occulter Experiment", NASA MSFC, technical report, 1985. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19850021975.pdf [cited 05 May 2020]

[17] Whorton, M., "A Simulation of the Instrument Pointing System for the Astro-1 Mission", NASA MFSC, technical memorandum, 1991. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19910020747.pdf [cited 05 May 2020]

[18] Sharkey, J. P., "A TREETOPS Simulation of the Hubble Space Telescope-High Gain Antenna Interaction", Workshop on Structural Dynamics and Control Interaction of Flexible Structures, NASA MSFC, April 22-24, 1986. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19870013302.pdf [cited 08 May 2020]

[19] Kim, Y. K., "Design and Analysis of Precise Pointing Systems", NASA MSFC, technical report, 2000. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20000038224.pdf [cited 5 May 2020]

[20] Kim, Y. K., "Determination and Control of Optical and X-Ray Wave Fronts", NASA MSFC, technical report, 1997. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19970019706.pdf [cited 5 May 2020]

[21] Nurre, G. S., "A TREETOPS Simulation of the STABLE Microgravity Vibration Isolation System", NASA MSFC, technical report, 1999. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19990021252.pdf [cited 5 May 2020]

[22] Kim, Y. K., "Equations of Motion for the g-LIMIT Microgravity Vibration Isolation System", NASA MSFC, technical report, 2001. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20010109677.pdf [cited 5 May 2020]

[23] Kim, Y. K., "Development and Integration of Control System Models", NASA MSFC, technical report, 1998. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19980040093.pdf [cited 5 May 2020]

[24] Wells, E. M., and Francis, R., "Performance Evaluation of the Gravity Probe B Design", NASA MFSC technical report, 1996. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19970016440.pdf [cited 5 May 2020]

[25] Kim, Y. K., "Modeling the Multi-Body System Dynamics of a Flexible Solar Sail Spacecraft", conference paper, Joint Army Navy NASA Air Force (JANNAF) Spacecraft Propulsion Subcommittee Joint Meeting, 2005. URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20060004763.pdf [cited 5 May 2020]

[26] Barhorst, A. A., "On modeling the dynamics of hybrid parameter multiple body mechanical systems." (1992): 4929-4929.
[27] Barhorst, A. A., and Everett, L. J., "Modeling hybrid parameter multiple body systems: A different approach", International Journal of Non-Linear Mechanics, Vol. 30, No. 1, 1995, pp 1-21.
[28] "LAPACK - Linear Algebra PACKage", URL: http://www.netlib.org/lapack/ [cited 05 May 2020]
[29] Intel Math Kernel Library, URL: https://software.intel.com/content/www/us/en/develop/tools/math-kernel-library.html [cited 8 May 2020]
[30] "The SPICE Toolkit," NASA Jet Propulsion Laboratory, California Institute of Technology, URL: https://naif.jpl.nasa.gov/naif/toolkit.html, [cited 1 May 2020]
[31] Spencer, J. L., "Pines' Nonsingular Gravitational Potential Derivation, Description and Implementation," McDonnell Technical Services Company, Inc., Houston Astronautics Division, prepared for NASA, contract NAS 9-13970, 1976.
[32] Tapley, B., Ries, J., Bettadpur, S. et al., "GGM02 – An Improved Earth Gravity Field Model from GRACE", Journal of Geodesy, 2005.
[33] "IERS Conventions (2010)", URL: https://www.iers.org/IERS/EN/Publications/TechnicalNotes/tn36.html [cited 1 May 2020]
[34] Chan, D.T., et al., "Space Launch System Booster Separation Aerodynamic Database Development and Uncertainty Quantification," AIAA, SciTech, 54th Aerospace Sciences Meeting, San Diego, 2016.
[35] Pinier, J. T., et al., "Space Launch System Liftoff and Transition Aerodynamic Characterization in the NASA Langley 14- by 22-Foot Subsonic Wind Tunnel" AIAA, SciTech, 53rd Aerospace Sciences Meeting, Kissimmee FL, 2015.
[36] Smith, O. E., and Weidner, D. K., "A Reference Atmosphere for Patrick AFB," NASA MSFC, technical memorandum X-53139, URL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19640023655.pdf [cited 1 May 2020]
[37] "Earth Global Reference Atmospheric Model 2010 (Earth GRAM 2010)," NASA Technology Transfer Program, URL: https://software.nasa.gov/software/MFS-32780-1 [cited 1 May 2020]
[38] Heritage, L., "New Doppler Radar Wind Profiler Will Replace Old System Near Shuttle Landing Facility," URL: https://www.nasa.gov/exploration/systems/ground/new_doppler_radar.html [cited 1 May 2020]
[39] "Model-View-Controller," URL: https://en.wikipedia.org/wiki/model-view-controller [cited 30 April 2020]
[40] MATLAB®, Version R2016b, The MathWorks, Inc., Massachusetts, 2016. http://www.mathworks.com [cited 30 April 2020]
[41] Python, Versions 2.7 and 3.0, Python Software Foundation, Oregon. http://www.python.org [cited 30 April 2020]
[42] Van der Walt, S., Colbert, C., and Varoquaux, G., "The NumPy Array: A Structure for Efficient Numerical Computation", Computing in Science & Engineering, Vol 13, 2011, pp. 22-30. http://dx.doi.org/10.1109/MCSE.2011.37
[43] gnuplot homepage, URL: http://www.gnuplot.info [cited 20 April 2020]
[44] Hunter, J. D., "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, Vol. 9, 2007, pp. 90-95. https://doi.org/10.1109/MCSE.2007.55 [cited 20 April 2020]
[45] Schroeder, W., Martin, K., and Lorensen, B., The Visualization Toolkit: An Object Oriented Approach to 3D Graphics, 4th ed., Kitware, New York, 2006.
[46] "pickle - Python object serialization", URL: https://docs.python.org/3/library/pickle.html [cited 30 April 2020]
[47] "THE HDF5 Library & File Format", ULR: https://www.hdfgroup.org/solutions/hdf5/ [cited 05 May 2020]
[48] "PQP - A Proximity Query Package," URL: http://gamma.cs.unc.edu/SSV [cited 1 May 2020]
[49] URL: https://www.agi.com/home [cited 05 May 2020]
[50] Hanson, J. M., and Beard, B. B., "Applying Monte Carlo Simulation to Launch Vehicle Design and Requirements Verification", Journal of Spacecraft and Rockets, Vol. 49, No. 1, January-February, 2012.
[51] VanZwieten, T., "Time and Frequency-Domain Cross-Verification of SLS 6DOF Trajectory Simulations", 37th Annual AAS Guidance and Control Conference, Breckenridge, Colorado, 2014.
[52] McCarter, J. W., "Simulating Flights of Future Launch Vehicles and Spacecraft", NASA Tech Briefs, 2007, [online] URL: http://www.techbriefs.com/component/content/article/ntb/tech-briefs/software/1182 [cited 30 April 2020]
[53] "Program to Optimize Simulated Trajectories II (Post2)", URL: https://post2.larc.nasa.gov [cited 30 April 2020]
[54] "Apache Subversion", URL: https://en.wikipedia.org/wiki/Apache_Subversion [cited 30 April 2020]
[55] Stoneking, E., "Implementation of Kane's Method for a Spacecraft Composed of Multiple Rigid Bodies", AIAA 2013-4649, 2013.