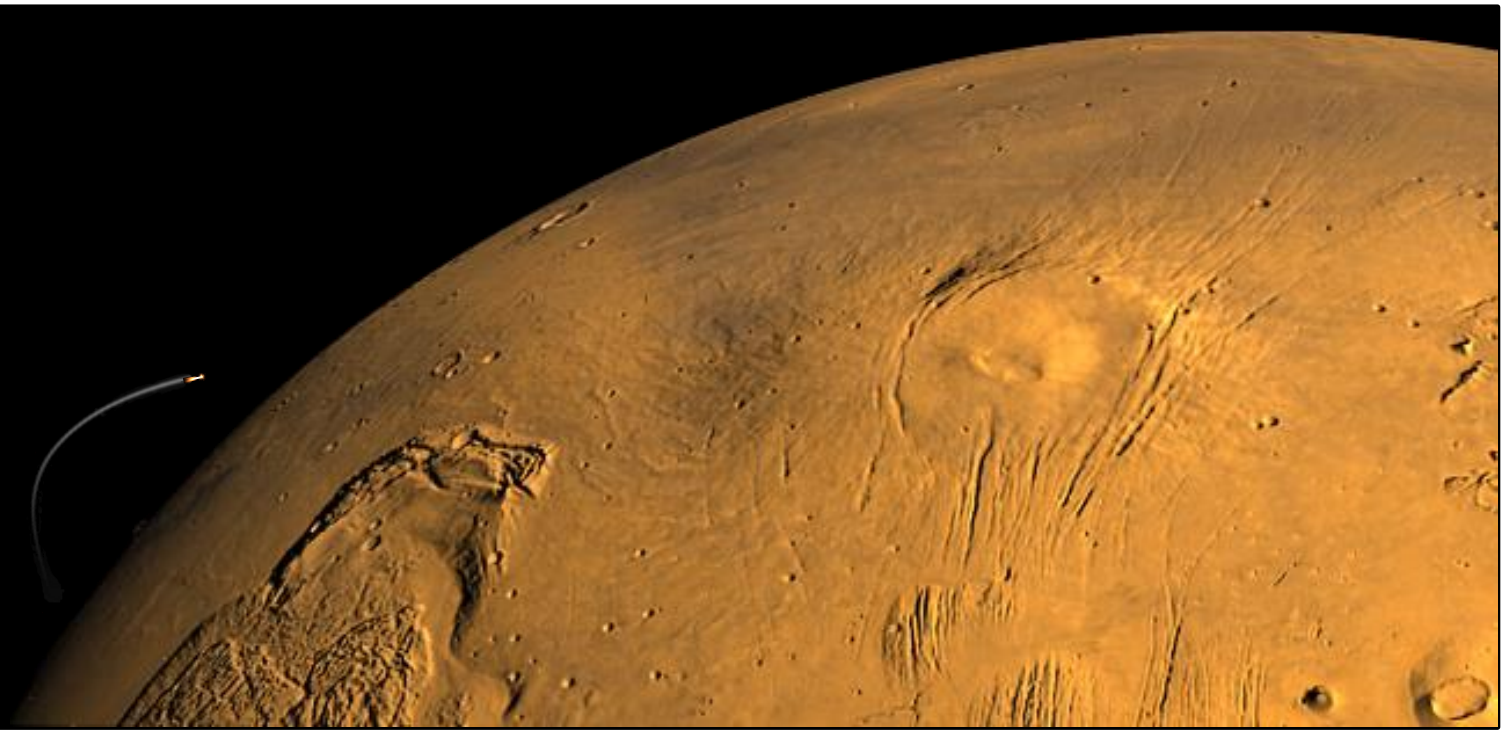
Mars Ascent Vehicle



Streamlining GNC Architecture Development and FSW Integration for the Mars Ascent Vehicle

Dorek Biglari
Jason Everett
Mike Fritzinger
Alex Summers

Introduction

- **Background**
  - The Mars Ascent Vehicle (MAV) is the ascent vehicle portion of the Mars Sample Return (MSR) campaign, and is expected to launch to the Martian surface no earlier than 2026
  - MAV has gone through several different vehicle configurations that require fast-turnaround, reliable, and modular GNC algorithms
  - Simultaneous GNC development and FSW trades require a unique collaboration between the two teams
  - V&V considerations for code must span both internal GNC algorithms and GNC-FSW interaction layers
  - Specific components of the GNC subsystem are baselined for auto-code, while others for direct-code
  - MAV GNC must be capable of complete remote configuration from Earth while verifying successful flight before FSW uploads

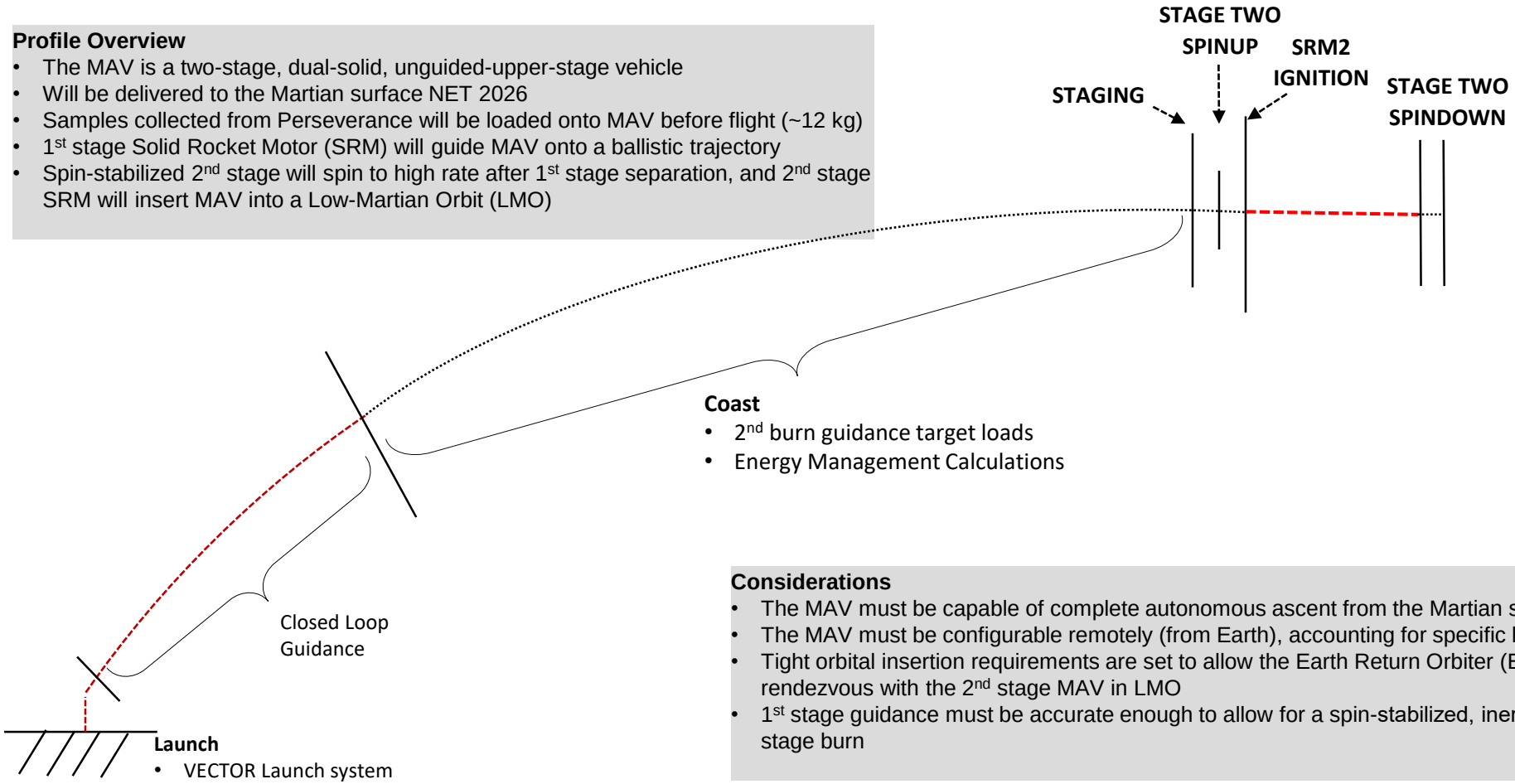
- **Solutions**
  - The MAV GNC team has worked with the MAV FSW team to coordinate fast and reliable code sharing
  - A lean, agile development approach has been adopted by the two teams in order to remain resilient to quick turnaround requests
  - Auto-coding, static analysis, unit testing, and V&V methodologies have been explored via both open-source toolkits and Mathwork's recommended auto-code workflows

- **Presentation Overview**
  - MAV mission profile
  - GNC & FSW subsystem overview
  - MAV GNC development processes and pipelines
  - V&V methodologies for MAV GNC
  - GNC auto-coding trades
  - Future work

MAV Mission Profile

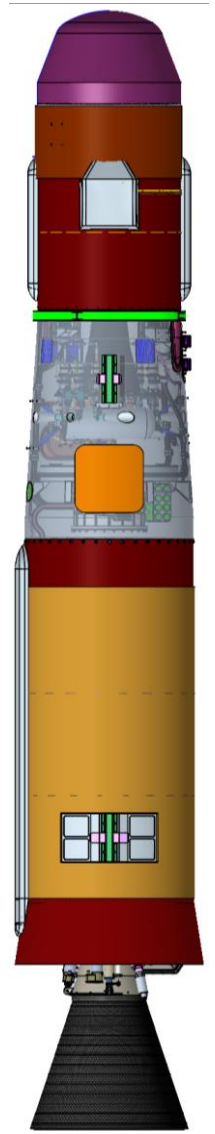
National Aeronautics and Space Administration
Marshall Space Flight Center
Mars Sample Return / Mars Ascent Vehicle

Profile Overview
- The MAV is a two-stage, dual-solid, unguided-upper-stage vehicle
- Will be delivered to the Martian surface NET 2026
- Samples collected from Perseverance will be loaded onto MAV before flight (~12 kg)
- 1st stage Solid Rocket Motor (SRM) will guide MAV onto a ballistic trajectory
- Spin-stabilized 2nd stage will spin to high rate after 1st stage separation, and 2nd stage SRM will insert MAV into a Low-Martian Orbit (LMO)



Considerations
- The MAV must be capable of complete autonomous ascent from the Martian surface
- The MAV must be configurable remotely (from Earth), accounting for specific landing state
- Tight orbital insertion requirements are set to allow the Earth Return Orbiter (ERO) to rendezvous with the 2nd stage MAV in LMO
- 1st stage guidance must be accurate enough to allow for a spin-stabilized, inertially “fixed” 2nd stage burn

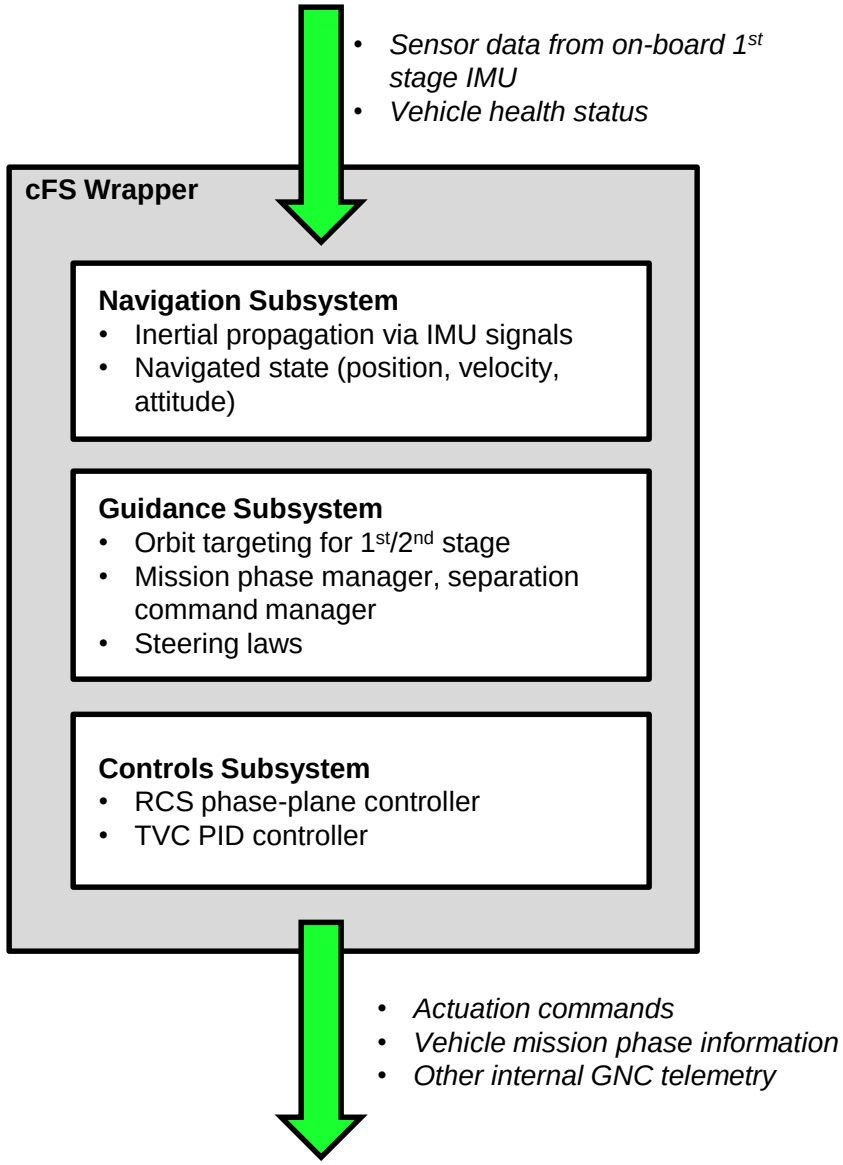
MAV Vehicle Configuration



GNC & FSW Subsystem Overviews

- **GNC for MAV**
  - Guidance and Navigation subsystems are both hand-coded in C++11, while the Controls subsystem is auto-coded into C++ via MATLAB
    - Trades are ongoing for best approach for this configuration (later in presentation)
  - Navigation is un-aided, IMU only
  - Multiple different phases of flight require different operational modes for GNC:
    - Pre-launch GNC activation routines
    - TVC steering w/ RCS roll-channel support
    - RCS phase-plane steering during coast

- **FSW for MAV**
  - cFS-based architecture employed for MAV FSW
    - Allows for table-based remote configuration for all software subsystems from Earth
  - Lightweight, low-power board requires efficient and computationally inexpensive code
  - Software-In-the-Loop (SIL) testing planned to test FSW-compliant versions of GNC code
    - Validation eventually to be performed with Hardware-In-The-Loop (HWIL)
  - Independent simulations tested in FSW group to verify GNC behaviour

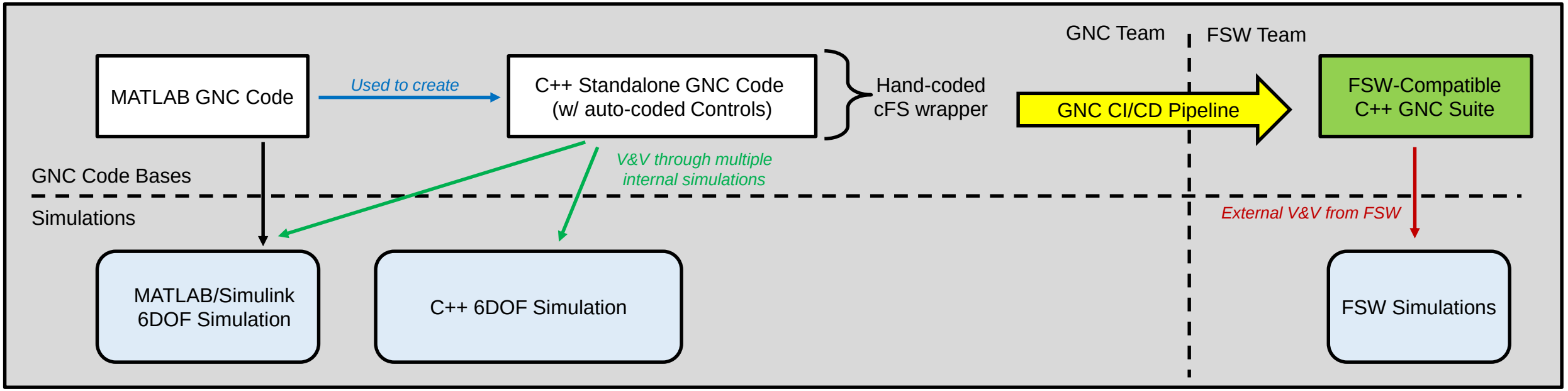


MAV GNC Development Process

- **Development Overview**
  - MAV GNC is currently being developed in a MSFC-housed MATLAB/Simulink 6DOF simulation suite
  - C++ 6DOF simulation being developed for V&V
  - A full MATLAB/Simulink model, and C++ standalone model, are both maintained containing GNC algorithms to be used for MAV
    - C++ standalone also contains a MEX file for testing C++ standalone code in MANTIS

- **Relation to FSW**
  - An agile approach is taken between the two teams with transparent code sharing on both sides
  - Synchronized Continuous Integration/Continuous Deployment (CI/CD) between two teams allows for compatible unit testing & static analysis methodologies

Overview of GNC Code Bases & Simulation Suites

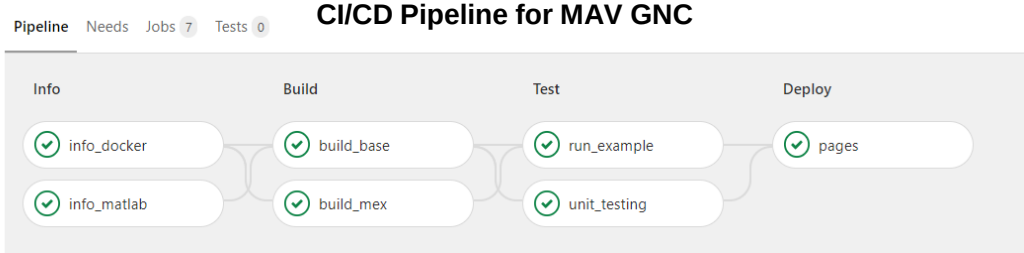


V&V Methodologies for MAV GNC

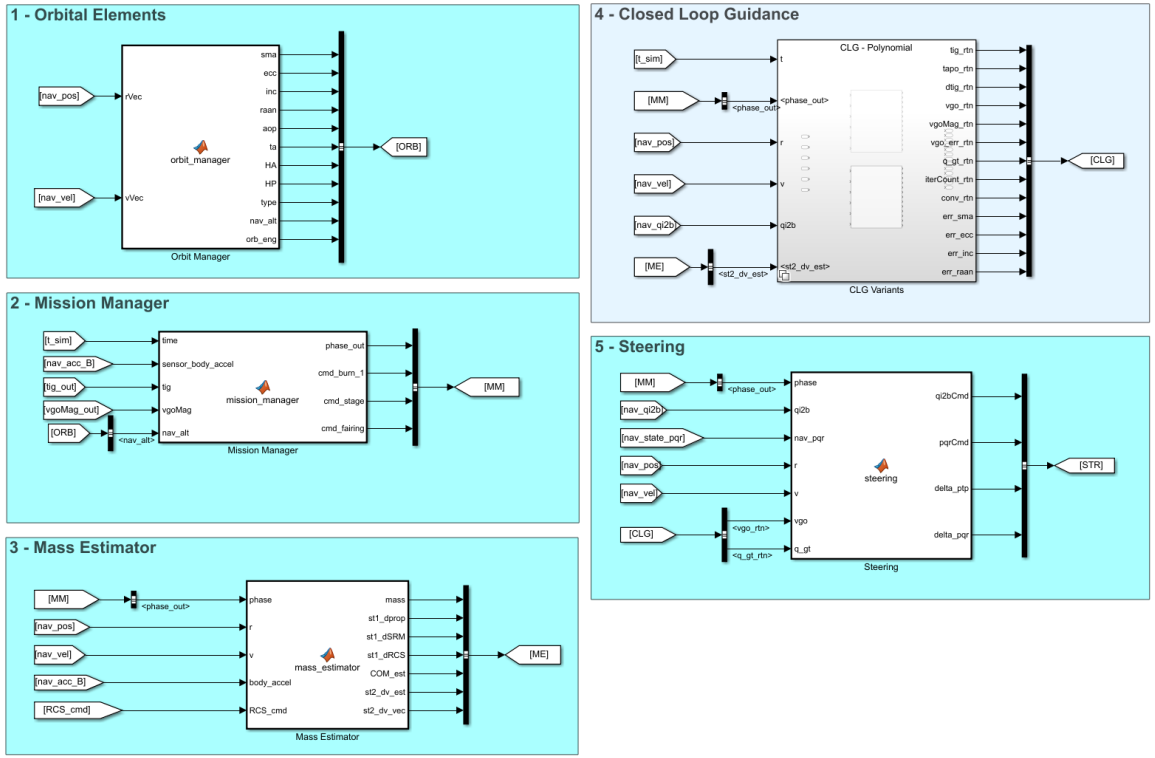
National Aeronautics and Space Administration
Marshall Space Flight Center
Mars Sample Return / Mars Ascent Vehicle

- **Dual maintenance of GNC Code**
  - By maintaining both C++ and Simulink code bases, the MAV GNC team can more easily break down different GNC components by functionality with specified inputs/outputs
  - Design by modularized functions allows for easier unit test generation, easier code flow analysis, and a stronger connection between auto-coded GNC and hand-coded GNC
  - Inputs/modifications flow both from Simulink to C++ and from C++ to Simulink when new findings, issues, or code changes come into play
- **CI/CD Pipeline**
  - GitLab CI/CD employed for MAV GNC C++ code to run unit tests, automatically deploy documentation, run verification routines on code, etc.

CI/CD Pipeline for MAV GNC



Simulink Guidance Subsystem



Modularized approach to GNC development allows for easy verification of submodules designed both in Simulink and in C++

C++ Guidance Subsystem

```
/**
 * Main guidance executable function, called each timestep.
 *
 * This function acts as the main executable process of the guidance subsystem, which
 * progresses through all functions required to successfully operate the guidance subsystem.
 */
void guid_exec(guid_config *GC, guid_ref *GUID)
{
    /* Calculate orbital parameters. */
    orbit_manager(GC, GUID);

    /* Run the mission manager. */
    mission_manager(GC, GUID);

    /* Run target manager. */
    target_manager(GC, GUID);

    /* Run mass estimator. */
    mass_estimator(GC, GUID);

    /* Calculate open-loop first stage commands. */
    open_loop_commands(GC, GUID);

    /* Run closed-loop first-stage guidance logic. */
    SxS(GC, GUID);

    /* Run TIG. */
    tig_manager(GC, GUID);

    /* Run steering logic for output. */
    steering(GC, GUID);

    return;
};
```

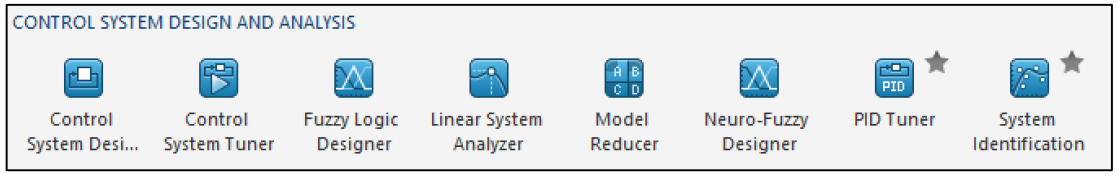
GNC Auto-Coding Trades

National Aeronautics and Space Administration
Marshall Space Flight Center
Mars Sample Return / Mars Ascent Vehicle

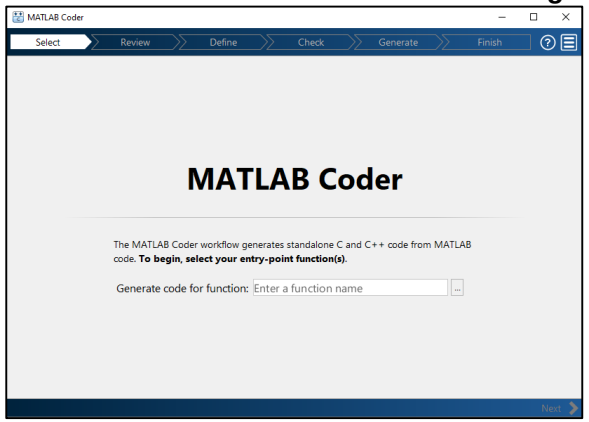
- **Trade Overview**
  - Original configuration of direct-coded guidance/navigation and auto-coded controls fueled by powerful MATLAB/Simulink control system design tools not available in C++ simulations
  - MAV GNC team is currently conducting trades on several different GNC configurations for potential delivery to FSW
  - Working with Mathworks teams to gain familiarity w/ static analysis, unit testing, and auto-coding methodologies to ensure produced code is NPR 7150.2C compliant
  - Each configuration to be tested in two-week agile sprints, with quantitative scoring of configurations developed from performance metrics:
    - Code readability
    - Code compatibility with simulations & FSW
    - NPR 7150.2C compliance
    - Code speed

- **Trade Outcome Benefits**
  - Multiple different GNC configurations allow for V&V of our functional models through several different simulations and code generation techniques
  - Mathworks unit testing & static analysis tools can be used on direct-coded software as well, so insight can be gained by results of using these tools on both direct-code and auto-code

MATLAB Apps for Control System Analysis



MATLAB Coder for C-based auto-coding



MAV GNC Tradespace

Navigation	Guidance	Controls	cFS Wrapper	Notes
Simulink Auto-Code	Simulink Auto-Code	Simulink Auto-Code	Simulink Interface Layer (SIL)	Full auto-coded GNC suite using SIL to generate a cFS wrapper for the code
Simulink Auto-Code	Simulink Auto-Code	Simulink Auto-Code	Direct-Coded Wrapper	Full auto-coded GNC suite with a hand-coded cFS wrapper
Direct-Code	**Direct-Code**	**Simulink Auto-Code**	**Direct-Coded Wrapper**	**Current configuration for MAV System Requirements Cycle (SRC)**
Direct-Code	Direct-Code	Direct-Code	Direct-Coded Wrapper	Full direct-coded GNC suite

Conclusions

Future Work

- The MAV GNC team plans to continue to work in-tandem with FSW team to support rapid generation and V&V testing of the GNC suite in both simulation and FSW-ready configurations
- GNC auto-coding trades to be complete around SRC close-out (August 2021)
- GNC functional documentation
  - Documentation to be included w/ Doxygen code documentation
  - Algorithm descriptions, algorithm derivations, functional inputs/outputs
  - More formalized methods (UML diagrams, etc.) can be used for functional flow

Questions?

Acknowledgements

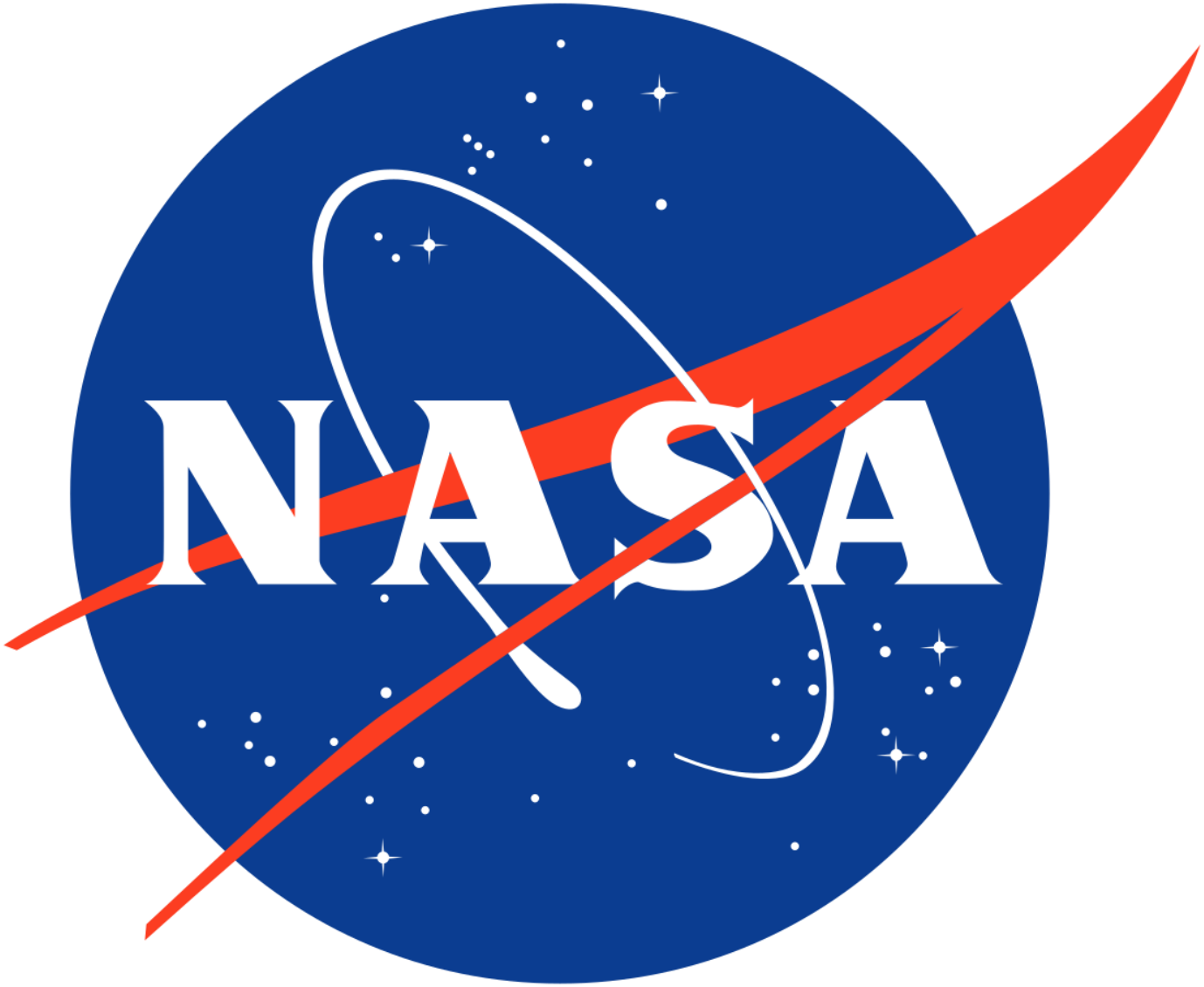
MAV GNC Team Lead
Joey Powers

MAV GNC Team
Evan Anzalone
Brian Bae
Dane Erickson
Jason Everett
Mike Fritzinger
Joey Hakanson
Alex Summers
Curtis Zimmerman

MAV FSW Team
Dorek Biglari
Stefanie Justice
Jason Stelly

MSFC EV42 GNC Team Lead
Andy Heaton

MSFC EV42 Branch Supervisor
Paul Von der Porten



References

- Anzalone, Erickson, Everett, Powers. “Guidance and Navigation Challenges for a Mars Ascent Vehicle”, *IEEE Aerospace*
- Everett, Jason, “A Generalized Guidance Approach to In-Space Solid-Propellant Vehicle Maneuvers,” *AAS GNC 2020*

Abstract

The Mars Ascent Vehicle (MAV) will be the first vehicle to perform an ascent from the surface of another atmospheric planetary body outside of the Earth-Moon system. Significant light-time delay requires complete autonomy of flight throughout ascent, and naturally a high level of reliability is desired in both MAV's hardware and software subsystems. The MAV Guidance, Navigation and Controls (GNC) team and the MAV Flight Software (FSW) team have partnered together to improve the efficiency of algorithm integration onto the MAV flight processor, and to increase confidence that said integration is successful and without human error. An interface architecture is proposed for the GNC suite that allows both the guidance and navigation subsystems to provide code algorithms directly in C++, and the controls subsystem to provide MATLAB Simulink auto-coded algorithms. Several continuous integration/deployment (CI/CD) methodologies have been considered for ease of transition of algorithm code from the GNC team to the FSW team. The GNC/FSW teams also worked together to develop a cFS-friendly wrapper which abstracts the integration of the GNC algorithm code into an interface-level API that is compatible with cFS. Several iterations of vehicle GNC code have been produced between the GNC/FSW team's partnership, and this strong interface between these two teams have allowed the GNC/FSW teams to greatly increase confidence of efficient and error-free implementation of the GNC code onto MAV for a successful flight.