

# SSim: NASA Mars Rover Robotics Flight Software Simulation

Vandi Verma, Chris Leger  
Mobility and Robotics Systems Section  
NASA Jet Propulsion Laboratory California Institute of Technology  
4800 Oak Grove Drive, Pasadena California, USA  
vandi@jpl.nasa.gov, leger@google.com

*Abstract*—Each new Mars rover has pursued increasingly richer science while tolerating a wider variety of environmental conditions and hardware degradation over longer mission operation duration. Sojourner operated for 83 sols (Martian days), Spirit for 2208 sols, and Opportunity is at 5111 sols, and Curiosity operation is ongoing at 2208 sols. To handle this increase in capability, the complexity of onboard flight software has increased. MSL (also known as Curiosity), uses more flight software lines of code than all previous missions to Mars combined, including both successes and failures[1]. MSL has more than 4,200 commands with as many as dozens of arguments, 54,000 parameters, and tens of thousands of additional state variables. A single high-level command may perform hours of configurable robotic arm and sampling behavior. Incorrect usage can result in the loss of an activity or the loss of the mission. Surface Simulation (“SSim”) was developed to address the challenge of making full and effective use of many capabilities of MSL, while managing complexity and risk. SSim is software that performs rapid context sensitive simulation of flight software. NASA Mars missions are comprised of three phases: several months of Cruise, a brief but exciting Entry Descent and Landing (EDL), and a Surface mission that typically lasts as long as the hardware survives. SSim is meant for use during the surface phase when the mission fulfills its primary objectives. The focus of SSim on MSL was the robotic flight software, including rover mobility and navigation, robotic arm manipulation, and sample acquisition, processing, and delivery. It can execute behaviors in simulation a thousand times faster than they execute in real time on the flight compute element. SSim is used by rover drivers to develop and validate command sequences throughout the planning cycle. SSim has been used to plan all of the Curiosity robotic operations since landing and is expected to continue to be used for the remaining life of the rover. Due to the impact of SSim on MSL, the Mars 2020 mission plans to increase the scope of SSim during flight operations, simulating not only rover planner operations, but all surface operations, including the instrument, power, thermal and telecommunication behavior. SSim is part of the Rover Sequencing and Visualization (RSVP) suite of Rover Planning tools (Yen, J. et al, 2005). In the paper we provide an overview of SSim architecture, design, implementation, and usage on MSL, as well as an overview of plans for Mars 2020.

## TABLE OF CONTENTS

1. INTRODUCTION.....	1
2. MSL ROVER .....	2
3. MSL OPERATIONS .....	3
4. ROBOTIC OPERATIONS.....	4
5. MOBILITY AND NAVIGATION .....	4
6. ROBOTIC ARM MANIPULATION .....	4
7. IN-SITU SAMPLING .....	4
8. SSIM ARCHITECTURE AND DESIGN .....	5
9. SSIM CAPABILITIES.....	7

978-1-5386-6854-2/19/\$31.00 ©2019 IEEE  
The contributions of co-author Chris Leger were completed between 2009-2013 when he was an employee of JPL, Caltech

10. CONCLUSIONS .....	9
ACKNOWLEDGMENTS .....	10
REFERENCES .....	10
BIOGRAPHY .....	11

## 1. INTRODUCTION

Mars science is an active area of research, and there are 500 planetary scientists that participate in MSL operations. Analysis and discussion of observations made by the rover typically impact the goals for subsequent observations. Unlike most deep space and Earth orbiting missions for which operations are typically planned weeks, months and even years in advance, Mars rover operations are planned daily during Mars time operations. Most NASA JPL Mars missions operate on Mars time for the first three months after landing[2], during which Earth operators align their schedules to a Martian day, of 24 hours, 39 minutes, and 35 seconds. As operations transition to a more sustainable range of working hours on Earth, 1 to 3 Sols of rover operation are planned each day, other than for special cases such as Mars solar conjunction. There are additional constraints such as Deep Space Network scheduling and time of Mars orbiter flybys over the rover that further limit the time available from when data first arrives on Earth to when the next plan must be uplinked to the rover on Mars.

A big factor in determining the scope of the plan for any given sol is plan complexity and the ability of human operators to guarantee that the plan is safe and meets the science and engineering intent in the limited time available. SSim provides a capability for human operators to quickly check if their intent is correctly captured prior to execution. SSim has made it tractable for operators to develop plans with scope and complexity greater than any previous planetary rover mission.

SSim enables high confidence in the plan generated on a tight timeline by using actual flight software code initialized with current rover state. This enables faithful simulation of subtle flight software state interactions and potentially emergent behavior. What differentiates SSim from other software simulations is that it uses actual flight software instead of a simplified model, and that the design focus is not on testing and development, but on operations. Speed, determinism, interfaces for environmental feedback, replicating current rover state, and high rate visibility into predicted state are therefore given priority. This does not preclude the use of SSim during development and testing. SSim was also used during MSL Systems Integration and Test, ALTO, and Robotic Arm and Sampling Verification & Validation to simulate most sequences prior to execution on hardware. Due to the impact of SSim on MSL, the Mars 2020 mission has increased the scope of SSim during flight operations, simulating not only

robotic operations, but all surface operations, including the instrument, power, thermal and telecommunication behavior.

SeqGen[3][4], as implemented for rover missions, produces many false positive and warnings, and many errors are not in scope. In addition, it takes tens of minutes of assess command sequences for complex plans.

WSTS (Work Station Test Set) is a software simulation used by missions for testing during development. It runs on a VxWorks emulator and simulates Flight Software. It includes models of low-level interactions such as 1553 bus traffic. It is valuable for testing FSW. However, it has limited interfaces for environmental feedback and can only run about six times real time and does not currently have the capability to initialize to current rover state, limiting its use for rapid operations.

Rover Analysis, Modeling and Simulation (ROAMS) [5] is a physics based dynamic simulation. Its focus is on analysis, design, development and testing of robots. SSim could for example be interfaces to a simulation like ROAMS for sensor and environmental feedback. There are additional multi-body dynamics simulations such as Automated Dynamic Analysis of Mechanical Systems (“ADAMS”) [6].

Mobility Mechanics Modeling Toolkit (M3Tk) [7] contains basic kinematics, inverse kinematics, dynamics and inverse dynamics capabilities for articulated multi-rigid body systems. A research project created a model of MSL that included modeling the rovers inertia, dynamics and wheel-ground contact in M3Tk and provided feedback to SSim. Although the current M3Tk based MSL simulation takes tens of minutes to run, optimizations are ongoing. There are additional approaches for high fidelity wheel terrain interactions[8][9]. These approaches could all be used to provide alternate sensor and environment feedback to a SSim simulation.

Gazebo is a 3D rigid body simulator for robots. It is designed for software testing via dynamic simulation of robot responses to software stimulus via ROS (Robot Operating System) [10] interfaces. The same ROS interfaces can be used to control real hardware and the simulated robot in Gazebo. SSim could be interfaced to Gazebo for visualization and environmental feedback via a ROS interface in a manner similar to the RSVP visualization to which it currently interfaces.

SSim has influenced a number of additional custom software in the loop simulations since its development in 2009 and Mars operation starting 2012.

## 2. MSL ROVER

Figures 1 and 2 show the MSL rover, which is a 6-wheel rocker-bogie suspension vehicle with 4 independently steerable corner wheels. It has a 2.1m long robotic arm with five degrees of freedom (DOFs) that deploys two turret mounted instruments and 3 mechanisms - a 4 DOF rotary percussive drill to collect powered samples from rocks (Drill), scoop for soil collection (CHIMRA Scoop), 4 DOF sample processing and delivery mechanisms that can sieve and prepare portions and deliver them via actuated inlets to instruments (CHIMRA 1mm and 150um sieves, Portioner, Scoop, Tunnel, and Thwackers), and a brush for removing dust from surfaces (Dust Removal Tool). The rover is 3m long with its robotic

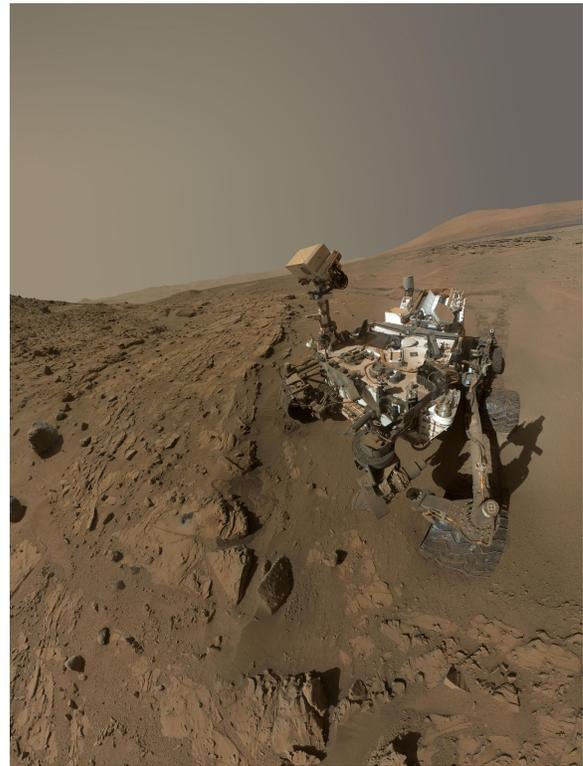


Figure 1. MSL Self-Portrait beside Windjana Drill holes on Mars.

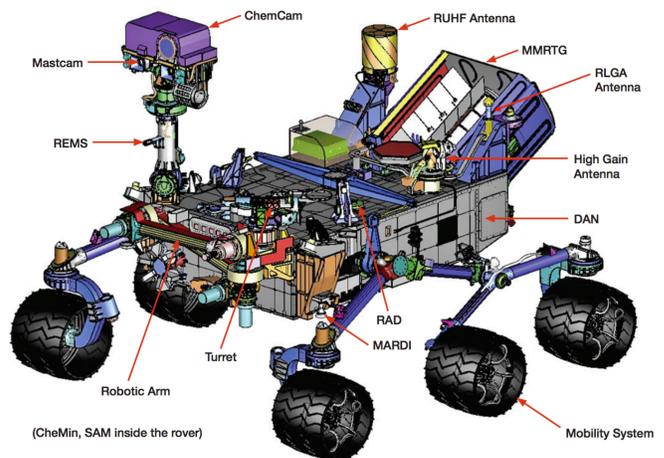


Figure 2. Simplified MSL rover.

arm stowed, 2.7m wide and 2.2m high at the top of its mast. Navigation makes use of stereo cameras, including wide-angle (120°) field of view (Hazzcam) cameras mounted in the front and rear of the chassis, and a pair of stereo navigation cameras (Navcams) with a field of view of 45° on the mast. With additional science imagers, 17 cameras are used by Curiosity. It weight 899kg and carries a payload of ten science instruments totaling 75kg - Alpha Particle X-ray Spectrometer (APXS), Chemistry and Camera (ChemCam), Chemistry and Mineralogy (CheMin), Dynamic Albedo of Neutrons (DAN), Mars Descent Imager (MARDI), Mars Hand Lens Imager (MAHLI), Mast Camera (MastCam), Radiation Assessment Detector (RAD), Rover Environmental Monitoring Station (REMS), and Sample Analysis at Mars

SAM).

### 3. MSL OPERATIONS

MSL is a science-driven mission. Mars science is an active area of research, and the science team revises its hypotheses and updates its goals for subsequent observations based on science discussions and analysis of the latest data from Mars.

Considerable simplification of rover commanding can be achieved by commanding in an event-driven manner. This is the approach used by MSL activities and is based on the prior Mars Exploration Rover sequencing model. The sequencing language allows nested conditional if-then-else branching on spacecraft state and time. There are 16 sequencing engines to run parallel sequences. The Master sequence controls the overall execution of the plan with the determinism of fixed allocations that are not event-driven.

MSL tactical operations is based on MER operations [2] and described in [11]. Operations consist of a series of working meetings briefly summarized below to discuss, develop and validate the plan, between which different roles perform their work.

#### *Strategic Planning*

The strategic planning process defines the long-term plan for the mission. Mars exploration missions have high-level science goals. Guided by these the science team develops strategic mission plans of varying horizon [12]. The time horizon of strategic plans may be on the order of months or years. Strategic targets can be identified from orbital images weeks in advance. However, as higher resolution surface imagery and data are available from the rover the shorter horizon plan may change. The daily planning of activities that respond to the data from the previous sol is referred to as tactical operations [2]. The Supratactical planning process bridges the gap between strategic and tactical planning. It maintains a detailed plan on the horizon of a week or month to ensure the constraints over a span of the science campaign are planned.

#### *Look Ahead Planning*

MSL has a large suite of instruments and sampling capabilities that are deployed to get correlated measurements on targets at a location of high science value. Look-ahead planning is the core of the Supratactical process. It folds in the variations from the tactical process into the short-term horizon, optimizing dependencies, resource usage, and constraints. Look-Ahead planning is conducted daily in parallel with tactical planning. The upcoming look-ahead schedule is revised based on new information.

#### *TACT*

The Tactical Activity Coordination Tagup is the first meeting in the tactical planning day. It occurs soon after the science and engineering teams have made a preliminary assessment of the data received from Mars. New engineering constraints for the plan are summarized, such as power, bandwidth, and communication pass times. The science team outlines the science activities under consideration for the plan guided by the Look-ahead plan. Following this, the science and engineering teams work on plan fragments, or abstractions of each activity. This determines which activities can feasibly fit within the constraints. At this stage, rover planners work closely with the science team and may run SSim interfaces

dozens of times. These include checking arm configurations on desired targets, visualizing clocking of tools and instruments with respect to terrain features, stamping the footprint of the brush area on the surface, and using the rover collision model to visualize the interaction of the robotic arm turret with the terrain.

#### *SOWG*

At the Science Operations Working Group meeting, the team evaluates whether the combined fragments of planned activities fit within the overall predicted resource constraints for the plan, including time, energy and data volume. If constraints are violated, changes are made to the plan, which may include reducing the scope of activities or removing them from the plan. By this time, rover planners must have a rough draft of expected commanding, including a reasonable estimation of duration of activities. This impacts the usage of resources and whether activities can complete in the time they've been allocated. Rover planners derive durations by running SSim on draft sequences. A given Sol's activities can be implemented in different ways that affect the duration of the activity. For instance, a single change in arm configuration could add as much as three minutes in movement duration and trigger a requirement to close and re-open an instrument cover, adding another five minutes. Since science observation sequences at specific arm positions may not be ready at this time, there is a mechanism for adding context-sensitive placeholder duration. One example of the sensitivity of execution to absolute time is the effect of rover body and terrain shadows, which can adversely affect science and engineering images. The evaluation of shadows with SSim-based tools can inform where in the plan the rover planner sequence is placed. Alternatively, activities within a long rover planner sequence may be reordered, or the rover planner sequence may be interleaved with other science observations.

#### *APAM*

At the Activity Plan Approval Meeting, a visualization of the activities is presented, and a final check of resources is performed. A poll is taken from the various engineering and science teams responsible to approve the aspects of the plan for which they are responsible. The visualization is generated by running a SSim simulation. Prior to this, rover planners typically run dozens of simulations and revise their commands based on SSim simulation results. More recent SSim-based tools such as ArmSketch automatically run dozens of parallel SSim simulations to evaluate constraints and optimize the trajectory of arm motion.

#### *Master / Submaster Walkthrough*

The team reviews select tactically developed sequences that implement the activities in the plan. The visual pose and context of the rover at each command is shown. This is based on SSim callbacks and library calls at the start and end of each command.

#### *Sequence Report Walkthrough*

SSim and rp-check [4] are run to perform a validation check of the final rover planner sequence. Rp-check statically checks the command sequence against a database of rules and is equivalent of programming language analyzers like lint [13]. Subsequently a combined validation check of the integrated sequences is performed. On MSL, this is done via Seqgen [3], which is based on SSim-provided input for rover planner sequences. On M2020, the current plan is to perform this via SSim as well. At the Sequence Report Walkthrough,

the team reviews results and dispositions unexpected violations.

#### CAM

At the Command Approval Meeting, a final review of all the files to be uplinked to the rover is performed.

There are additional activities performed for strategic and supra-tactical planning[12] for which SSim is also typically used.

### 4. ROBOTIC OPERATIONS

Rover drivers (also known as rover planners) are the human operators that control robotic arm, mobility, and sampling on Mars. This includes navigating the rover from one location to another, commanding the robotic arm to position all robotic arm-mounted instruments and tools, operating sample collection, processing, and delivery mechanisms, and handling all the coordination constraints, safety checks and state.

### 5. MOBILITY AND NAVIGATION

Rover drivers are responsible for generating the sequence of commands to safely navigate the rover from one location to the other. This may consist of precision drives to position targets in the arm workspace for a variety of constrained observations in a sampling campaign, or a hundred meter drive navigating around sand traps and craters. Rover drivers select between different methods for driving the rover depending on the terrain, their ability to resolve it in imagery, and the short- and long-term science intent.

#### *Directed driving*

Directed driving is also called “blind” driving and is used to command the rover to drive a certain distance over terrain that can be manually evaluated as safe. Images in the drive direction from previous rover positions are used to generate a 3D terrain mesh and feed this into a visual simulation[14][15][16] evaluate the terrain. No on-board terrain evaluation and navigation is performed for such drives, which are therefore fastest. Reactive checks of rover state, such as constraining suspension and attitude, may still be performed to stop the drive.

#### *Visual Odometry*

Visual Odometry (VO) is used for more accurate position estimation. The rover tracks features between image pairs to compute the vehicles six degree-of-freedom position and attitude change based upon observed relative displacement of features. Depending on the terrain and the accuracy needed for science and safety, rover drivers choose the amount of VO to perform. It may be performed at every step, for select waypoints, or periodically, such as in sandy terrain to ensure the rover is not spinning its wheels or digging into sand.

#### *Autonomous Navigation*

Autonomous Navigation (AutoNav) takes stereo images, evaluates hazards and then selects drive paths. This requires on-board terrain analysis, and on the 133 MHz RAD750 processor, when AutoNav is performed in conjunction with VO, autonomous drive rates are almost six times slower than those when driving blind. However, it allows the rover to safely navigate over terrain that is unknown to operators. A



**Figure 3. MSL Robotic Arm and Turret.**

variation of AutoNav called “guarded” mode permits directed motion over terrain that is unknown to operators, which only executes if AutoNav evaluates the path to be safe.

### 6. ROBOTIC ARM MANIPULATION

Figure 3 shows the five degree-of-freedom robotic arm, which is used for positioning turret-mounted instruments and tools with respect to Mars surface and rover-mounted targets.

Figure 4 shows the turret-mounted tools and instruments. Rover drivers must assess and manage clearances between the turret and terrain. Millimeter-level positioning accuracy is needed from the 100kg arm and 0.6m diameter turret, which is 33kg of the weight. The quality of the science data collected from the turret-mounted Alpha Particle X-ray Spectrometer (APXS) and Mars Hand Lens Imager (MAHLI) instruments can depend on the accuracy of placement on a feature of interest. In addition, these instruments can easily be damaged by collision with the terrain if placed incorrectly. For example, the MAHLI lens cannot be allowed to contact the surface, yet must regularly be placed within centimeters of it for the best science data. The Dust Removal Tool (DRT) brushes can be bent if not spinning when moved to contact the surface. Robotic arm flight software compensates for sources of error including deflections, backlash, and thermal distortions.

### 7. IN-SITU SAMPLING

MSL is the first rover to perform in situ sampling on Mars. It is designed to acquire rock and regolith samples from the surface, sieve them to acquire particles that are less than 150um or 1mm and deliver small portions via actuated inlets into SAM and Chemin instruments. SAM analyzes chemistry, including carbon chemistry, and Chemin uses X-ray diffraction to study mineralogy. The sample must then be cleaned from the system and a visual inspection of the turret performed prior to the next sampling. In order to perform

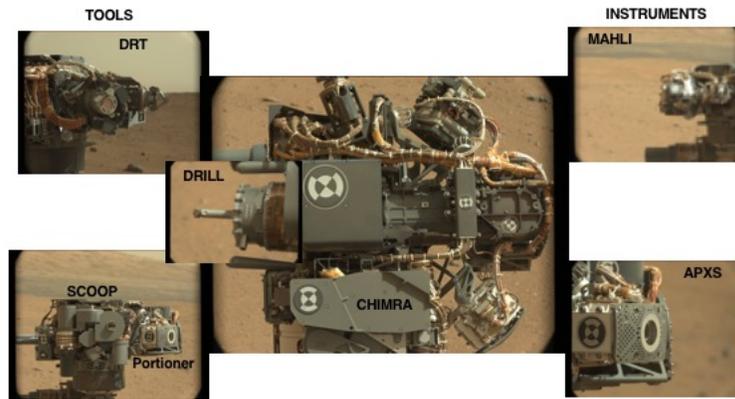


Figure 4. MSL turret showing instruments and tools.

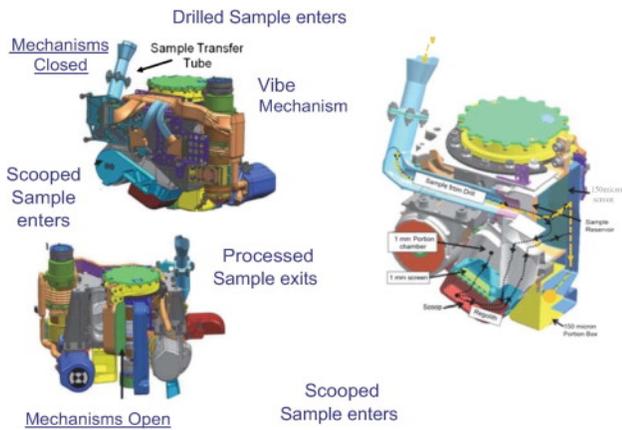


Figure 5. Sample pathways through the CHIMRA tool on the MSL turret.

its functions, the Sample Acquisition/Sample Processing and Handling (SA/SPaH) system consists of 17 actuated degrees of freedom (four on the drill, four on CHIMRA, one on the DRT, five on the Robotic Arm, and three on the Inlet Covers). Some actuators have resolvers, and there are force sensors and contact switches. These 17 degrees of freedom are used in a carefully choreographed manner to perform sampling operations.

The properties of Mars samples was unknown prior to landing. Commanding had to be designed to evolve with the mission. The motions are therefore composed of commands that are reconfigurable. Figure 6 shows a flowchart of the high level sampling commands and the various pathways that may be used tactically depending on science and engineering goals. The arm must reposition Drill and CHIMRA with respect to gravity for sample flow. Therefore, the specific arm trajectory varies with the rover attitude, which changes with the terrain. Some of the CHIMRA sample flow pathways that must be achieved by coordinated motion of the Robotic Arm and CHIMRA joints are shown in Figure 5. There is risk of hardware damage if these are executed incorrectly. The exact motions can vary based on rover state. Rover Planners are responsible for ensuring safety for the robotics components

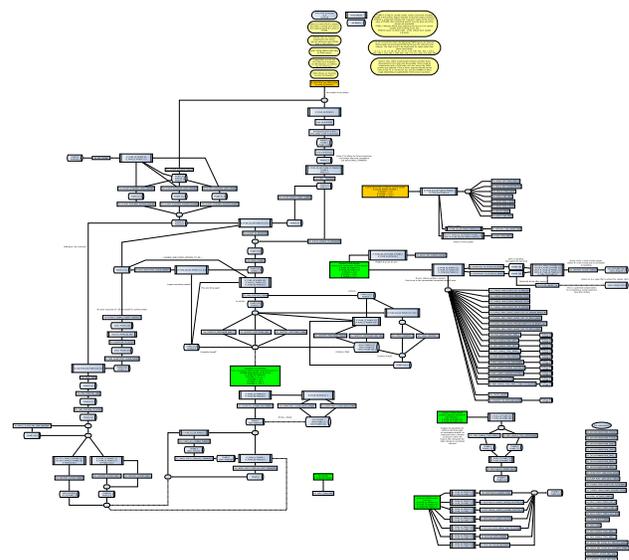


Figure 6. Flowchart of high level sampling commands.

at all times.

## 8. SSIM ARCHITECTURE AND DESIGN

Flight software follows a rigorous development and review process. The software is extensively tested at various levels from unit tests, to Flight software Integrated Tests, Systems Integration and Tests with hardware, Verification & Validation at various levels, and Assembly Launch and Test Operations. However, MSL flight software is one of the most complicated aspects of the spacecraft. This is reflected by a quote from the MSL project manager prior to landing on Mars:

*“I’m not worried that the radar will not perform. We’ve tested the hell out of that, and we got good performance off the radar. I’m not worried that the engines are not going to fire. I’m not worried that the parachute’s not going to inflate, but I am worried that there’s a bug in the software that we*

*haven't caught yet, and that we don't know about, and it will come and bite us on a bad day.”[17]*  
-MSL Project Manager, Pete Theisinger

### *SSim Guiding Principles*

SSim design was based on three guiding principles:

- Use actual Flight Software code to simulate subtle Flight Software state interactions and emergent behavior.
- Make execution fast, portable, and repeatable by abstracting all hardware interfaces to allow running entirely on Linux. Abstract any flight software modules where modeling a subset of flight software behavior is sufficient.
- Replicate on-board state by using telemetry received on ground to initialize simulation.

Simulating flight software in large part as-is allows rover drivers to predict with high accuracy behavior in a given context. If the behavior deviates from expectation it allows us to compensate for it. There is complementary value in predicting expected modeled behavior as well. For this MSL rover drivers use rp-check in addition for static sequence checking[4].

### *High-level MSL flight software architecture*

MSL flight software architecture is based on the MER architecture. As described in [18] and [19]. Flight Software is a collection of modules. The design emphasizes interfaces, encapsulation and modularity between modules. Modules are arranged in layers for which the lowest layer interfaces with hardware and the highest layer encodes the behavioral logic. Flight software runs on the VxWorks real time operating system. It consists of multiple preemptive prioritized tasks (threads). Not all modules have tasks. Some modules may be libraries. Other modules have multiple tasks such as the seq module which implements the sequencing capability and has 16 tasks, one for each sequence engine. Modules communicate via Inter Process Communication (IPC) messages, which are implemented via VxWorks pipes.

Some core principles of the MSL flight software architecture, which are common to most robotic spacecraft developed by NASA Jet Propulsion Laboratory, are as follows[19]:

1. Modules communicate asynchronously via messages
2. Each task executes an event loop which processes the arriving messages.
3. The task waits only on message arrival, at only one point in the code, and acts on the message according to the modules top-level state machine. One of more Finite State Machines are regularly used to process messages.
4. All modules are initialized before any module is activated.
5. Flight software autocode generates module initialization, command parsing, message handling functions
6. Multi-threaded concurrent with real time constraints
7. A finite state machine module initializes and activates all other modules, spawns tasks and manages redundant computers.

### *High-level SSim Flight Software Simulation Architecture*

Flight software runs on-board the rover on Mars and has real time constraints. It is multi-threaded and non-deterministic. For purposes of providing feedback to rover drivers, the simulation needed to be repeatable, and hence deterministic. It needed to be hundreds of times faster than real time since taking an entire day to simulate a full Sol's plan was not useful. Rover drivers typically run many dozens of simulations

during an average planning cycle. The SSim architecture takes advantage of the flight architecture to make simplifying assumptions. In SSim as in flight software:

1. Modules communicate via IPC messages
2. Each task executes an event loop which processes the arriving messages.
3. The task waits only on message arrival, at only one point in the code, and acts on the message according to the modules top-level state machine. One of more Finite State Machines are regularly used process messages. Most of the surface robotics module use Hierarchical State Machines (HSMs)[20].
4. All modules are initialized before any module is activated.
5. Flight software autocode generates module initialization, command parsing, message handling functions
6. Multi-threaded concurrent with real time constraints
7. For repeatable determinism, SSim is Single-threaded concurrent since it doesn't have real time constraints. SSim initializes and activates all modules without spawning tasks.

Like Flight Software, SSim is written in C/C++ with flight software coding standards. On MSL, it is part of the flight software repository but is not part of the flight binary. On M2020, SSim is in an independent repository and accessible to a suite of M2020 Mission System tools. It will run on GovCloud<sup>2</sup> to support these tools.

### *SSim Input*

To initialize to match the state of the rover on Mars, SSim takes as input flight software state. SSim can also record the state resulting from sequence execution. This state can be used to initialize subsequent SSim simulations if multiple Sols are being planned back to back without intermediate data from Mars. Command sequences are input to SSim in Flight Software sequence binary format. The RSVP tool uses the Rover Markup Language (RML) for user interface and editing. The format allows users to provide additional simulation only options that are mapped to simulation-only Flight Software commands. The sequences are then converted to the Flight Software sequence format. Sampling scripts that provide the parameterized decomposition of high-level sampling commands as described in section 7, are also provided as input via files. RSVP provides feedback on the environment via rover settling, slip models, and terrain classification.

On MSL this ground derived reflection of on-board state is referred to as “NPM” since the state is stored in Non-volatile Parameter Memory flash on the rover. SSim based automated stand-alone tools are used to generate ground based NPM. These tools take as input prior state and new telemetry received on the ground and convert them to spacecraft and simulation only commands that are executed through SSim. By stubbing the interface flight software uses to write to on-board NPM memory SSim produces files replicating NPM.

### *Environmental feedback*

Planetary robots face the challenge of considerable variability from environmental interaction. There is high variance in the terrain MSL encounters on Mars and the targets and material with which the robotic arm and sampling systems interact.

<sup>2</sup>AWS GovCloud (US) is an isolated AWS region designed to host sensitive data and regulated workloads in the cloud, helping customers support their U.S. government compliance requirements, including the International Traffic in Arms Regulations (ITAR) and Federal Risk and Authorization Management Program (FedRAMP).

The simulation therefore provides callback functions to provide feedback, such as rover attitude and suspension settling on terrain. This terrain is built by generating a 3D terrain mesh by performing stereo processing on the images we get from Mars using stereo vision cameras. For autonomous navigation beyond the range of the navigation cameras, lower resolution localized images are used, which are taken from orbital cameras such as HiRise on the Mars Reconnaissance Orbiter.

Where sufficient information about the environment is lacking, such as force sensor feedback and weight on drill bit while drilling into a rock, a model is averaged from past flight data. This results in some rate of penetration, which may vary on Mars and is compensated by allocating margin for the allowed maximum duration for the activity. Planetary exploration involves daily robotic interaction with features no human has ever seen before, so bounding the expected hardness and adding fault protection in flight software is part of the design.

## 9. SSIM CAPABILITIES

SSim was developed to address the growing complexity and challenge of driving, operating the robotic arm, and sampling on Mars. With user input and environmental feedback from other RSVP components, some of the capabilities it provides are as follows:

### *Predict plan execution*

SSim predicts plan execution so rover planners can validate that no unexpected errors are present and that the behavior meets their intent. SSim will regularly trigger valid errors during the early stages of sequence development, and it would be unusual to get through a planning cycle without SSim having signaled an error. As described in Section 3, there are 16 sequencing engines that may be used during planning to issue commands in parallel. This includes activity constructs in which sequences ping-pong back and forth, invoking each other based on state triggers to implement a self-sustaining complex behavior. Rover drivers need to be able to visualize and inspect sequence execution to determine whether the plan is achieving the desired intent. An example of one of the types of high-level visualizations is shown in 7. The simulated path depends on parameters including the maximum step size between visual odometry (VO) positions, which cameras to use for VO, and the rover positions in which to take the images. There may be high level commands to stop for mid-drive imaging iteratively every few meters, or at a specific location. The lighting for the VO images may be impacted by terrain interactions and slip, since the drive through a sandy patch may take extra steps to compensate for slip and occur later.

### *Predict rover safety risks*

MSL is sometimes called the swiss army knife of rovers. It is capable of a wide range of activities. A large proportion of robotics commands also have high frequency environmental interactions. SSim simulations allow context-sensitive visualization of these commands. The joint angles for deploying the arm positioning the MAHLI imager within a centimeter of the surface may be safe in one context. In another context, or if the terrain were mis-modeled, the robotic arm could be permitted to collide with a terrain feature. Another example of hardware safety risk is described later in section 9 on SSim monitoring of cached sample.

### *Prevent loss of science*

Flight software is designed to ensure hardware safety. However, the safety check logic can trip in conditions humans know to be benign. This is aggravated when rover capabilities are used in new ways. SSim helps ensure they don't result in the loss of activities.

### *Predict state interactions*

Reconfiguration of low-level parameters and state may change high-level behavior and interaction. It is important to be able to predict what the rover will do not only at a high level, but its effects on state and environmental interactions that could have subtle consequences. Early in the mission, SSim modeled the MastCam only at an abstract level. However, when Curiosity wheels started showing excessive wear, high resolution MastCam images of the wheels were taken. Flight software will not permit any mobility if the focus mechanism is not in the stowed position. However, one drive also included VO for accurate position estimation, and the seed image was taken prior to first motion. VO has an efficiency option that allows skipping the extra imaging if driving is disallowed. A subtle state interaction is that the VO command unconditionally reinforces the state indicating whether driving is allowed. Since this wasn't a drive motion command, and there was no real risk to the MastCam, it went unnoticed and resulted in failing the entire drive that Sol. Following this, the scope of SSim was extended to include modeling the MastCam. Another example is the use of SSim to model state interaction from Chemcam LIBS firing. The Chemcam instrument on MSL is the first planetary science Laser-Induced Breakdown Spectrometer (LIBS). The LIBS instrument uses powerful laser pulses, focused on a small spot on target rock and soil samples within 7 m of the rover, to ablate atoms and ions in electronically excited states from which they decay, producing light-emitting plasma. The plasma light is collected by a 110 mm diameter telescope and focused onto the end of a fiber optic cable. It is a powerful science instrument that allows remote elemental analysis with no sample preparation. However, the operations team must ensure that the LIBS beam is not fired on any part of the rover itself which would result in a fault. The lib beam is modeled as part of the rover self-collision model and SSim is used to check if LIBS targeting in terrain relative frames may result in the LIBS laser intersect the rover body.

### *Predict environmental interaction*

Flight software state variables in SSim are initialized with telemetry from Mars that affect behavior. For example, the rover attitude can impact arm trajectories via deflection compensation, or if the arm end effector is being commanded with respect to gravity. This could impact the relative position of the arm with respect to the surrounding hardware and result in a collision. In other cases, RSVP is used to provide feedback on rover pose via a slip model. This can be used to detect excessive slip faults.

### *Speed*

During planning, rover drivers generate tentative drive and arm trajectory sketches and repeatedly simulate with SSim and inspect the outcome. Using this feedback, they can apply their full range of expertise to revise the plan to meet the high-level intent.

Humans, even domain experts, are not suited to mentally tracking hundreds of thousands of variables, but they are good at evaluating high-level intent.

To be useful for human operators to iteratively simulate and revise the plan on a tight timeline, the feedback needs to be fast.

#### *Options for simulating off-nominal paths*

Mars rover Flight Software is designed to put the rover in a safe state if a fault is detected. Historically, Mars mission operations simulate the nominal path and not any contingency commanding to account for uncertainty and faults. However, there are mechanisms for simulating off-nominal execution paths in SSim. Any conditional "if" statement can be forced to specifically take either the "then" or the "else" execution path. Typically, the execution of conditionals is based on evaluation of the conditioned expression given the rover state at that point in SSim execution of the sequence. Alternately, motion mechanisms have stop commands that will interrupt ongoing motion. There is a mechanism to issue commands via the RSVP toolset that are only sent in simulation and not in flight. This can be used to test off-nominal paths that interrupt ongoing motion, say of the robotic arm. A parallel test sequence is typically executed in SSim that triggers a stop. Various criteria such as absolute or relative time, or when a particular command in the sequence under test starts execution, may be used to issue the stop.

#### *Visual Odometry*

Visual Odometry ("VO") enables the rover to accurately track its position relative to ground-identified hazards and traverse goal locations in the presence of slip[21]. Although VO on MSL is considerably faster than on previous Mars rovers it still takes on the order of ~45sec per step, so it is used judiciously. In addition, it has constraints on number and distribution of features on which successful stereo correlation can be performed in the image pairs being compared. Pointing for VO images can be in any coordinate frame relative to the rover, or a fixed surface site frame. SSim provides hooks for feedback from slip models. VO flight software in SSim provides slip model estimated position delta if enabled. Hence, if the command sequence being developed does not perform VO updates, the rover tracks in simulation will deviate from expected surface slip tracks as the rover on Mars would be oblivious to real slip, and if VO is used the rover commanding in simulation and on Mars would compensate for drive direction slip. Figure 7 shows a drive with VO at Marias Pass on Mars simulated with SSim.

#### *Uncertainty*

One of the mechanisms in SSim for accounting for uncertainty is via margin or "halos." SSim halos are extra margin that can be added to safety checks when simulating. Commands sent to Mars do not include the margin. These halos allow the simulation to be robust to execution uncertainty and differences between on-board state and state available during simulation. Commands will trip Flight Software checks in SSim if the margin is insufficient. Some examples are rover tilt, rocker/bogie suspension, and yaw limits, margin that can be added to Keep-Out and Keep-In Zones which are rover planner designated bounding areas that the rover will not enter or exist respectively and extra margin that may be added to rover self collision checks. Another example is the halo SSim can add to the rover self collision model. Every robotic arm command sequence is required to pass SSim simulation with at least 3mm of self collision margin to allow for uncertainty between Earth simulation and Mars execution such as due to thermal impacts on hardware.

simulation. This makes it easy for rover drivers to get

feedback on why the commanded arm motion failed, for example due to failure to find an inverse kinematics solution, or joint limit, or collision, etc. This allows for fast targeted correction of the sequence.

#### *Frame transforms*

There are over a hundred coordinate frames that may be used for commanding. Some are rover body relative and other are surface relative. SSim uses frame transformation flight software to perform coordinate transforms similar to the vehicle on Mars. SSim allows quickly checking impacts on the remainder of the plan from changes in frames. For example, there are times when surface targets are modified later in the planning day when new information is available either from Mars or due to maturation of the plan. There are cases when modifying the target frame results in a Chemcam LIBS collision with rover hardware when there was none when firing LIBS at the previous target frame. Simulating via SSim detects such conflicts and allows for them to be resolved.

#### *Sampling*

For each of the configurable high-level sampling commands in Figure 5 SSim simulates the low level behaviors and state interactions for a rover planners to evaluate. The intent of the blurred figure is to communicate the scope and dependencies of the high level commands. Grey rectangles in the figure represent high-level commands where a single command performs behaviors such as "transfer drilled sample" or "prepare a sample portion". Each of the high level sampling commands shown can consist of as many as 200 lower level commands. To provide a sense of the scale of mechanism motion for sampling, the arm Rover Motion Counter (RMC) in the first MSL sampling site frame at Rocknest alone ended up at 4808. As it increments at the start and end of each arm move, single joint or compound movement of the arm was commanded 2404 times. By comparison, at the end of the nearly yearlong journey Opportunity made through Victoria Crater, operators had commanded movement of the arm (or Instrument Deployment Device) 2303 times. The number of via points for an average low-level MSL arm move is much larger, and this doesn't take into account CHIMRA actuation. In addition to the tactical usage of SSim, it was also used extensively for the development of the high-level sampling behaviors that must be evaluated for sampling and other safety constraints for the range of environment and rover state in which they may be used. One example is the usage of SSim for computing tilt robustness of the high level sampling commands by running tens of thousands systematic simulations over the tilt cone. In operations rover planners use tilt plots similar to these to guide final position during drive approach to a sampling site, or to select between different configurations for sample dropoff. Figure 8 shows one such example of a resulting tilt plot.

#### *Mast mounted cameras*

On MSL, the mast has two degrees of freedom for nominal operations usage. Imaging commands can point mast mounted cameras in a variety of co-ordinate frames. The mast is used for Visual Odometry, Autonomous Navigation, Chemcam observations, and other science and engineering imaging. At times coordinated arm and mast motions are used for turret inspection or for inspecting hardware on the mast. SSim is used to develop and validate these to ensure that the imaging intent is met, the pointing is safe, and the field of view is adequate.

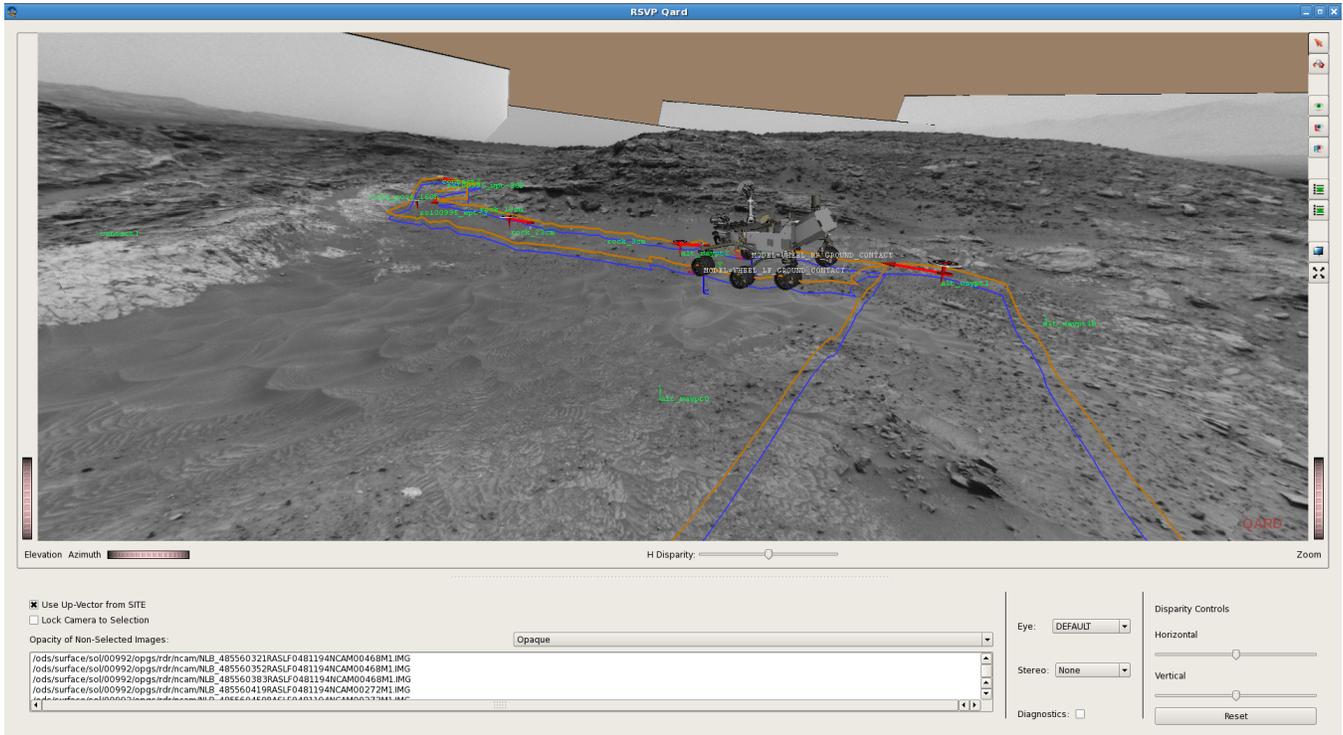


Figure 7. SSim simulation of VO drive in RSVP.

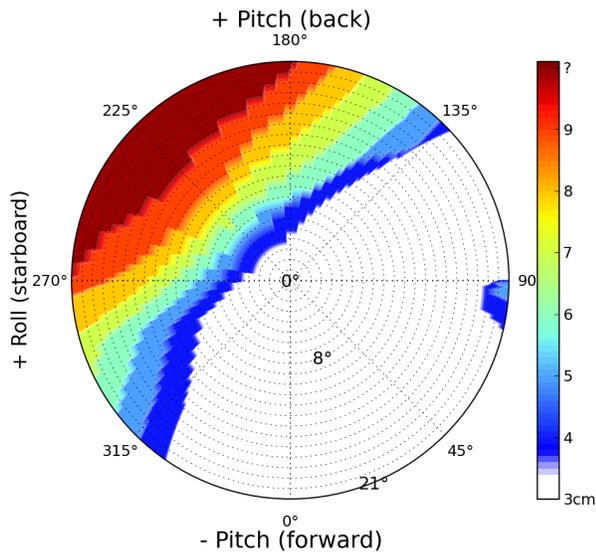


Figure 8. SSim based tilt plot showing collisions in red for dropoff to SAM1 via the 150um sample processing pathway.

#### Extending Flight Software Capability

After the first drilling operation at John Kline, the science team requested the capability to cache the sample and drive so that subsequent portions could be generated from that sample to deliver to SAM, while still allowing driving to alternate locations and using the arm for contact science along the way. SAM can run different variants of its analysis for each delivery. Updating on-board flight software is a long process, and so SSim was updated to monitor the turret

gravity vector and track 34 sampling states. If constraints that can potentially damage sampling hardware are violated, SSim generates an error. Figure 9 shows MAHLI cached sample contact science using SSim monitoring. Additional details on the cached sample capability are provided in[22].

## 10. CONCLUSIONS

High speed simulations such as SSim that use the actual Flight Software can provide a powerful analysis, testing and operations capability. They remove the need to replicate in models the behavior of increasingly complex Flight Software. The determinism of the simulation makes it valuable for integrated software regression testing.

SSim has fulfilled the need for predicting the behavior of semi-autonomous systems. It has been used for every MSL robotic operations plan since landing in 2012.

Future missions and robotic systems can benefit from following this approach. Regardless of the level of autonomy, there is eventually human intent behind robotic systems. SSim provides a capability for human operators to quickly check if their intent is correctly captured by the robot prior to execution. In addition, its speed makes it an effective regression testing tool for simulating with a variety of different initial and intermediate states and environmental feedback. It provides the ability to directly alter and query Flight Software state that may not be exposed via commands and telemetry. It can also be used to sample a solution space to ensure robustness.

Mars 2020 has decided to extend SSim usage and use it for the validation of all sequences in the integrated plan for the sol including instruments and infrastructure. In addition the operations team plans to reduce the amount of review

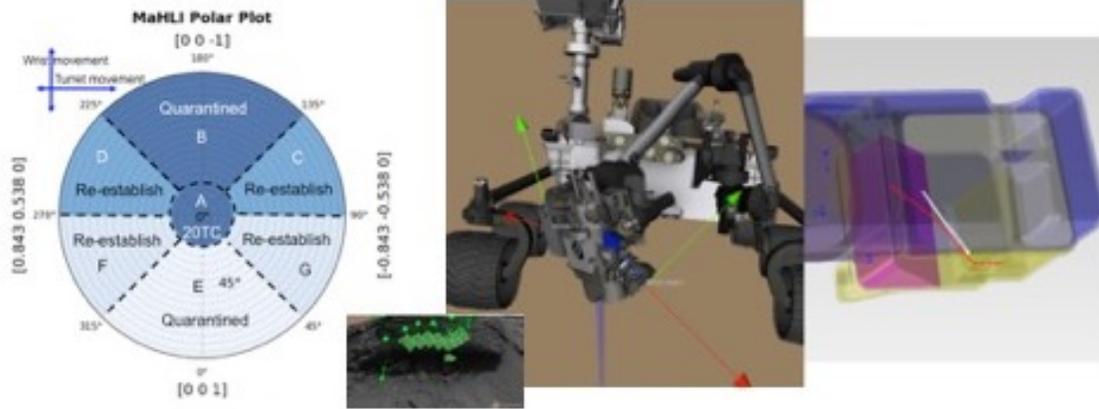


Figure 9. MAHLI contact science with SSim cached sample monitoring.

and allow the simulation to highlight the failures to bring to attention. Mars 2020 SSim regression tests often inform Flight Software developers of bugs during development.

### ACKNOWLEDGMENTS

The research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. Several MSL and Mars 2020 flight software team members, in particular Joseph Carsten, Todd Litwin, Dan Helmick, Won Kim, Mark Maimone, Dan Gaines, Jeff Biesiadecki and MSL RSVP development team members, in particular Jeng Yen, John Wright, Frank Hartman, Steven Myint, Nick Wiltsie, Scott Maxwell and Brian Cooper have contributed to it in many ways. Thanks to MSL Engineering operations team, in particular James Borders, Liz Duffy, Megan Richardson, and Esteben Rodriguez, Arm and sampling systems engineering team in particular Matt Robinson and Stephen Kuhn, MSL rover drivers, and tactical operations teams for feedback and suggestions. Thanks to MSL and M2020 projects, specially Jim Erickson, Alicia Allbaugh, Andy Mishkin, and Jennifer Trosper for supporting this work. Thanks to the Mars 2020 SSim team, specially Steven Myint, Daren Lee, Kris Wehage, Luis Fischer, Trevor Reed, Viet Nguyen, Jeng Yen, Jay Torres, Usha Guduri, Jim Kurien, Rachel Collins, Collette Lohr, Brian Roth, Nick Rossomando, Sean Wenzel, and Jigna Lad for pushing the envelope and for new things to come.

### REFERENCES

- [1] G. Holzmann, "Mars Code," *Communications of the ACM*, vol. 57, no. 2, pp. 64–73, Feb. 2014.
- [2] A. H. Mishkin, D. Limonadi, S. L. Laubach, and D. S. Bass, "Working the Martian Night Shift: The MER Surface Operations Process," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 46–53, 2006.
- [3] A. Patrikalakis and T. O'Reilly, "Advances in discrete-event simulation for MSL command validation," Delft Netherlands, 2013.
- [4] M. W. Maimon, S. Maxwell, J. J. Biesiadecki, and S. S. Algermissen, "RP-check: An Architecture for Spaceflight Command Sequence Validation," 2018.
- [5] A. Jain, J. B. Balam, J. Cameron, J. Guineau, C. Lim,

- M. Pomerantz, and G. Sohl, "Recent developments in the roams planetary rover simulation environment." in *In Proceedings of 2004 IEEE Aerospace Conference*, Big Sky, Montana, Mar. 2004.
- [6] B. Trease, "Adams-Based Rover Terramechanics and Mobility Simulator ARTEMIS," 2013.
- [7] R. Mukherjee, S. Myint, J. Chang, I. Kim, J. Craft, M. Pomerantz, J. Kim, and L. Peterson, "M3tk: A Robot Mobility and Manipulation Modeling Toolkit." in *ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 7, Buffalo, New York, USA, Aug. 2014.
- [8] F. Zhou, R. E. Arvidson, K. Bennett, B. Trease, R. Lindemann, P. Bellutta, K. Iagnemma, and C. Senatore, "Simulations of Mars Rover Traverses," *Journal of Field Robotics*, vol. 31, no. 1, pp. 141–160, 2014.
- [9] J. B. Johnson, A. V. Kulchitsky, P. Duvoy, K. Iagnemma, C. Senatore, R. E. Arvidson, and J. Moore, "Discrete element method simulations of Mars Exploration Rover wheel performance," *Journal of Terramechanics*, vol. 62, pp. 31–40, Dec. 2015.
- [10] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, L. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System." 2009.
- [11] D. Gaines, G. Doran, H. Justice, G. Rabideau, S. Schaffer, V. Verma, K. Wagstaff, A. Vasavada, W. Huffman, R. Anderson, R. Mackey, and T. Estlin, "Productivity Challenges for Mars Rover Operations: A Case Study of Mars Science Laboratory Operations," Tech. Rep. D-97908, 2016.
- [12] D. Chattopadhyay, A. Mishkin, A. Allbaugh, N. Cox, G. Tan-Wang, and G. Pyrzak, "The Mars Science Laboratory Supratactical Process," Pasadena CA, 2014.
- [13] S. C. Johnson, "Lint, a C Program Checker," 1978.
- [14] R. Deen, D. A. Alexander, and J. Maki, "Mars Image Products: Science Goes Operational," May 2004.
- [15] D. Alexander, P. Zamani, R. Deen, P. Andres, and H. Mortensen, "Automated generation of image products for Mars Exploration Rover mission tactical operations," vol. 1, 2005, pp. 923 – 929 Vol. 1.
- [16] J. Wright, A. Trebi-ollennu, F. Hartman, B. Cooper, S. Maxwell, J. Yen, and J. Morrison, "Terrain Modelling

for In-situ Activity Planning and Rehearsal for the Mars Exploration Rovers,” in *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, 2005, pp. 1372 – 1377.

- [17] F. J. Moring, “Sky Crane. Aviation Week & Space Technology,” *Aviation Week & Space Technology*, pp. 38–44, Aug. 2011.
- [18] G. Reeves and J. Snyder, “An overview of the Mars Exploration Rovers flight software,” The Big Island and Hawaii, 2005.
- [19] K. P. Gostelow, “The Mars Science Laboratory Entry, Descent, and Landing Flight Software,” Kauai, HI, United States, Feb. 2013. [Online]. Available: <https://ntrs.nasa.gov/search.jsp?R=20150007480>
- [20] D. Harel, “Statecharts: A visual formalism for complex systems,” *Journal Science of Computer Programming*, vol. 8, no. 3, pp. 231–274, Jun. 1987.
- [21] M. Maimone, Y. Cheng, and L. Matthies, “Two Years of Visual Odometry on the Mars Exploration Rovers,” *Journal of Field Robotics*, vol. 24, no. 3, pp. 169–186, Feb. 2007.
- [22] V. Verma and S. Kuhn, “Refactoring the Curiosity Rovers Sample Handling Architecture on Mars.” in *In Proceedings of the IEEE Aerospace Conference*, Big Sky, Montana, USA, Mar. 2019.

*simulation for the MER Rover Sequencing and Visualization Program (RSVP). Dr. Leger received a BS in Computer Engineering and MS and PhD degrees in Robotics from Carnegie Mellon University. He currently works at Google, Mountain View, CA.*

## BIOGRAPHY



**Vandi Verma** is the Operable Robotics Group Supervisor in the Mobility and Robotic Systems Section at NASA JPL. She designed and developed SSim for MSL. She had additional roles on MSL for the design and development of AEGIS autonomous targeting Flight Software, MSL Sample Processing Systems engineering, Flight Software Integrated Testing, Systems Integrated Test-

ing, and Motor Control Testing. Since 2008 she has been driving Mars rovers (MER, MSL) and operating the robotic arm and sampling system. She is currently working on Sample Caching Algorithms and Flight Software for the Mars 2020 mission. She has worked on a number of research robotics projects and deployed autonomous robots in the Arctic, Antarctica and Atacama. She has a Ph.D. in Robotics from Carnegie Mellon University. Her thesis was on particle filters for robot fault detection and identification.



**Chris Leger** Chris Leger was the Surface FSW Development Lead for the Mars Science Laboratory Mission, and a developer for the robotic arm and motor control interface flight software, and SSim. He previously worked as a rover driver and flight software developer for the Mars Exploration Rover (MER) mission, and as a mobility and motor control flight system engineer for

the Mars Science Laboratory mission. On MER, he wrote flight software for the Descent Image Motion Estimation System (DIMES), manipulator collision detection, and radar and pyrotechnic device interfaces. He also designed rover mechanical hardware, developed machine vision algorithms for the MER ground data system, and worked on manipulator