These Wheels Are Made for Arc-ing: Two New Mobility Commands to Improve Wheel Wear Outcomes

Mark Maimone
Jet Propulsion Laboratory,
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
mwm@jpl.nasa.gov

Neil Abcouwer
San Carlos, CA, USA
nabcouwer@gmail.com

PJ Rollins
Jet Propulsion Laboratory,
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
peter.j.rollins@jpl.nasa.gov

Evan Hilgemann
Jet Propulsion Laboratory,
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
evan.w.hilgemann@jpl.nasa.gov

Freddy Wang
Jet Propulsion Laboratory,
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
Jiun-Kai.F.Wang@jpl.nasa.gov

Nikunj Patel
Blue Origin
21650 Oxnard St., Suite 1555
Woodland Hills, CA 91367
nikunj.patel@blueorigin.com

Alexandra Holloway
Jet Propulsion Laboratory,
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
alexandra.holloway@jpl.nasa.gov

***Abstract*—The Mars Science Laboratory (MSL) Curiosity rover has driven over 29 kilometers since landing on Mars in August 2012, as of sol 3600 (September 2022). MSL Rover Planners use a variety of different driving commands and sequencing strategies to reach science targets of interest in diverse terrain. However, 490 Martian solar days (sols) and 4600 meters into the mission, the wheels began to exhibit significant wear, manifesting as cracks and breaks in their aluminum-alloy construction. Hardware mitigation was not possible due to the location of the vehicle; instead, engineers applied software techniques to extend the lifetime of the wheels. This initially led to the development of new Terrain-adaptive Wheel Speed Control software for the rover, which has proven helpful and now has been in use continuously since sol 1646 (April 2017).**

But some inefficiencies remained. When performing a precision approach to a goal location, the standard driving strategy alternates arcs (specified using a small number of fixed, preselected curvatures, running in open-loop motion) with closed-loop turns in place. But turn-in-place commands were found to exacerbate wheel wear rates due to their slower drive speeds and additional steering. So when a new flight software (FSW) release called R13 became possible, the project decided to implement two new commands that will help reduce unnecessary rover motion. These new mobility commands are now part of MSL's next major FSW release and offer new commanding strategies that should reduce unnecessary wheel wear and allow rover planners to generate more flexible drive paths.

While similar in name to existing arcing commands, these two commands have completely new behaviors that allow the rover to maneuver in ways never possible before: dynamically choosing the precise arc curvature appropriate for the precision goal location, and using the gyro-measured heading change to terminate motion. This update removes the prior restrictions on which specific curvatures can be used, letting the rover choose whatever is appropriate given its current pose relative to the current goal. These two commands can potentially replace the turn-in-place maneuvers currently used to reach a specific goal, a known significant contributor to wheel damage in terrains with embedded rocks.

The new commands were exercised during algorithm development both in simulation and also on MSL's Vehicle Systems Testbed (VSTB), an engineering model of Curiosity typically deployed in JPL's Mars Yard, an outdoor test area with a variety of terrains and slopes. Once development was complete, the VSTB was also used to conduct a formal Verification and Validation (V&V) test campaign over 5 days in the Mars Yard, testing the commands to their limits to ensure readiness for Mars. During this test campaign, the expected behavior of the commands was confirmed which led to approval for inclusion in the R13 FSW update after a review of the V&V results by a board of Mission Subject Matter Experts. We anticipate these commands will be deployed on Curiosity as part of the R13 update expected in early 2023.

TABLE OF CONTENTS

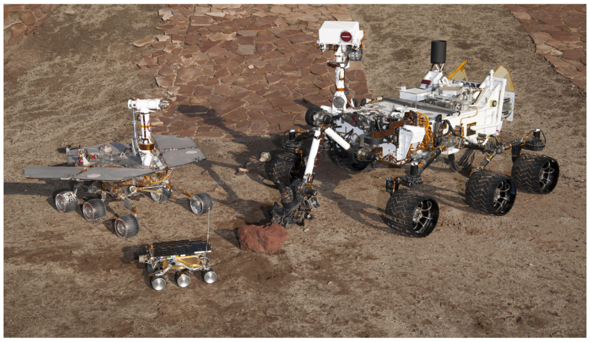


Figure 1: A Mars rover family photo: Opportunity (left), Curiosity (right), and Pathfinder/Sojourner (middle). This photo was taken in the Mars Yard outdoor test area.

Figure 2: Rover wheel wear: Martian wheel photos on sol 490 (left) and sol 2030 (right), with significantly more damage visible in the later photo

1. INTRODUCTION

Curiosity is a third-generation 6-wheeled Mars rover with a rocker/bogie suspension, following in the footsteps of Sojourner, Spirit, and Opportunity (see Figure 1 for a "family photo"). Curiosity has been exploring Mars since August 2012, and is likely to continue to operate for many more years.

An unexpected amount of wheel degradation was first noticed on Curiosity after 490 Martian solar days (sols) – nearly a year and a half of Earth time. Figure 2 shows a photograph taken by Curiosity's Mars Hand Lens Imager (MAHLI) instrument of one of the rover wheels on sol 490 (December 2013), highlighting the wear observed at the time, and an image of the wheel at the same pose from sol 2030 (April 2018) showing how the damage has increased over time.

The discovery of significant wheel wear led to several mitigation strategies, on different time scales.

At first, drives were restricted to be explicitly limited to at most 20 meters per sol, with human Rover Planners and Surface Properties Scientists required to characterize every step in the nearby terrain for wheel safety. But this approach was not tenable for long-term operations because of the limitations placed on how far the rover could travel, and due to the significant amount of overhead imposed on the the tactical operations team.

Next, the science team determined how to use orbital imagery to identify the most hazardous terrain for driving. This enabled the creation of new strategic drive paths that would avoid the most hazardous terrain. That strategy was indeed helpful, and led to greatly reduced wear rates [1]. While this approach helped us avoid mobility hazards in the short term when an alternate route existed, it did not address the fundamental problem of how to mitigate wheel damage, and indeed wheel damage was still occurring [4] [5].

Finally, strategies for updating the mobility flight software on the rover itself were considered. The initial effort was focused on low-level control of the drive motors during forward and backward motions in all drive modes. This led to the development of new terrain-adaptive wheel speed control, and after years of work this capability was uplinked to Mars in April 2017. This feature helps reduce forces as the wheels climb over small rocks or undulating terrain, and has been in use ever since [1]. However, flight software is configured to disable that mode when the rover turns in place.

A turn-in-place is implemented by steering the four corner wheels to approximately 45 degrees off-forward-axis, and driving them in the same clockwise or counter-clockwise direction. The center of the rover does not translate during a turn-in-place, except possibly due to slip. The onboard attitude estimate is maintained by integrating delta gyroscope measurements from the Inertial Measurement Unit at 8 Hz while the rover is turning. Motion terminates when the actual heading is about to reach the commanded heading. The corner wheels then must be steered back to continue driving along a forward or backward arc.

Detailed analysis by our engineering and science teams eventually led to the understanding that turns in place cause additional wheel damage, particularly to the middle wheels because they drive more slowly in that mode, which creates elevated loads when engaging obstacles [2]. Steering a wheel while sitting atop an obstacle at the tire edge also caused the highest loads observed during engineering tests, however due to the lack of specific damage features in flight this was determined to be a low-likelihood cause of the particular damage seen up to that point [3]. But turning in place is still commanded frequently, especially when attempting to reach a specific drive goal or sampling area.

So we looked for ways to reduce the amount of turning in place. The aspect of turning in place that makes it attractive for precision approaches is that it servos on the actual vehicle heading. That is in contrast to forward/backward arc motions, which are *not* based on sensed data, but rather use precomputed drive motor rotations in an open loop fashion, assuming a fixed wheel radius and no slip. So we considered developing new variations of arc commands that *would* incorporate sensor data. Doing so would allow us to remove many turns in place from noiminal drive plans, eliminating the circa 90 degrees of steering imposed on the corner wheels each time, and reducing the middle wheel damage caused by turns in place.

This paper describes the two resulting commands that were added to flight software to further mitigate wheel wear concerns, taking us into the second decade of Mars operations with a more capable set of mobility options. An earlier version of some of this information was provided in a presentation-only format [6].

2. BASIC MOBILITY

NASA's Mars rovers share a common heritage of mobility commands. The Mars Exploration Rover (MER) mission's

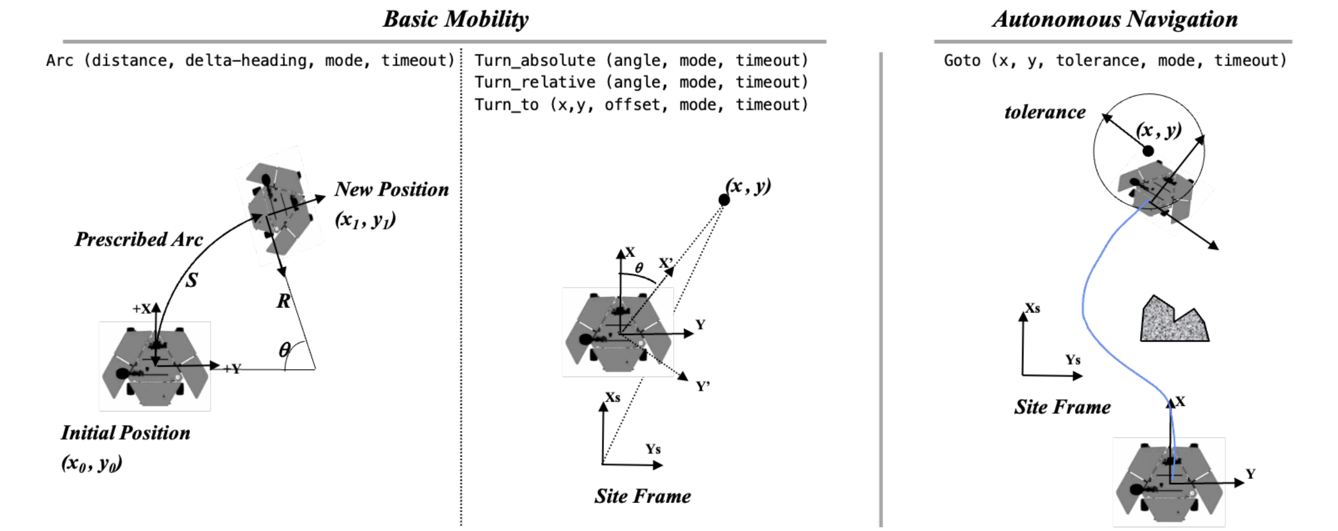


Figure 3: Two main types of MER heritage rover mobility: *Basic mobility arc* allowing motion along circular arc or straight line path of commanded length, open-loop relative to on-board position/heading estimate; *closed-loop turn* providing in-place turn about rover center to commanded heading, closed-loop around IMU sensor-based heading estimate; and *autonomous navigation* allowing traverse toward a commanded waypoint with on-board hazard detection using stereo vision; closed-loop around position and heading, but no fine positioning

Spirit and Opportunity rovers initially used a simple mobility commanding framework with two main types: basic and autonomous [7], depicted in Figure 3.

Basic mobility arc commands typically follow a pre-specified path relative to the current position. Commanding an arc must include a fixed distance and heading change specified in advance by human planners. But arcs are implemented as open-loop commands. They don't sense when the goal has been reached – they simply stop spinning the wheels when a pre-determined number of rotations have been achieved – resulting in limited confidence that the rover will reach the commanded position.

In contrast, basic turn commands do have closed-loop control. They stop turning when a sensor indicates that the rover is pointed in the desired direction. Turn goals can be specified using either relative or absolute headings, or as a particular ground goal location.

Finally, the autonomous mobility GO_TO command lets the rover choose its own path to a goal using current information about the terrain and rover state. However, the autonomous driving does not support fine positioning. Typically, the rover will stop some distance away from the drive goal, and operators will use basic commanding to reach the intended location.

This framework is also used on MSL, and has certain implications to wheel wear. Basic mobility arcs do not adapt well to unexpected slip. Autonomous GO_TO commands do, but they cannot do fine positioning. So to perform fine positioning, we begin to blur the lines between these commands.

3. Implementation of ARC_TO and ARC_UNTIL_TO

Operational needs on Mars eventually led the Spirit and Opportunity teams to add a new ARC_UNTIL mobility command (drive UNTIL you would pass the goal). This command looks a lot like an ARC command – operators still specify a distance and the heading change – but the command stops vehicle motion once the navigation goal location (or a goal line containing it) has been reached. Figure 4 illustrates this command.

In fact, using a TURN_TO command followed by this command became the preferred way to perform a precision approach to a particular drive location for both Spirit and Opportunity. Turning to face the goal and using several ARC_UNTILs to drive toward it remains the preferred strategy for Curiosity and Perseverance.

However, the ARC_UNTIL command only lets the rover adjust the drive distance, not the delta heading toward the goal. Such a change was considered for Spirit and Opportunity, but the extra complexity and uncertainty of allowing the rover to choose its steering led the team to choose not to implement it in a basic primitive drive command. Wheel wear considerations on Curiosity have forced a reconsideration of that decision.

The current precision approach strategy includes a turn-in-place followed by straight-line motion. That turn-in-place can add significant wheel motion, steering each corner wheel up to 90 degrees to enter a turn-in-place configuration and return to straight-ahead. Instead, allowing the rover to choose its own arc would minimize steering, and therefore reduce any forces generated by the turn-in-place.

We have thus implemented two new commands to decrease wheel motion:

The ARC_TO command calculates the delta heading argu-

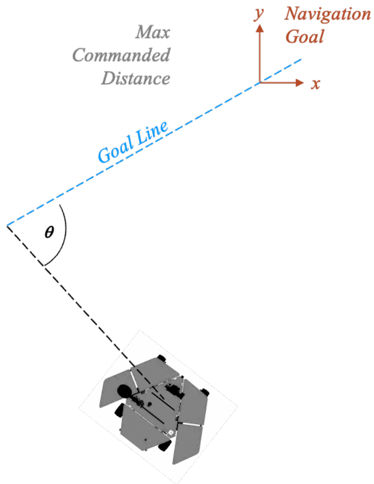


Figure 4: Illustration of ARC_UNTIL command inherited from the MER mission.

ment autonomously based on the relative location of the current navigation goal. It will then drive the rover toward that goal, without exceeding a specified path length. As a result, ARC_TO will minimize the steering change needed to reach a goal.

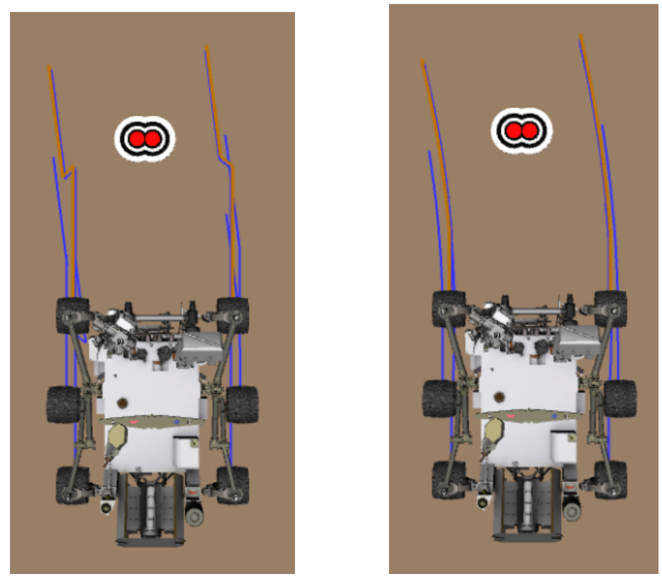
The ARC_UNTIL_TO command requires a particular arc to be specified, but will stop driving once the rover points at the current navigation goal, or meets other ARC_UNTIL stopping criteria. As a result, ARC_UNTIL_TO will give us high precision in pointing directly toward a goal. This is in contrast to the original basic arc commands which are all "open loop" and do not guarantee anything about the rover's final heading or position. ARC_UNTIL_TO measures the rover heading using the IMU sensor and therefore can help achieve precise pointing without ever requiring a turn-in-place configuration.

These new commands have been implemented in the R13 version of MSL flight software, which is expected to be deployed on Mars in early 2023 [8].

Simulation Analyses

In combination, ARC_UNTIL_TO and ARC_TO can replace current sequencing patterns and reduce wheel wear by substuting turn-in-place/ARC_UNTIL commands with smarter arc-based commands. These commands eliminate the middle wheel drive speed slowdown associated with turning in place, and reduce drive and steering motor usage. To demonstrate the reduction in motor usage, two canonical cases were simulated in JPL's Robot Sequencing and Visualization Program (RSVP) software package to quantify the effect of the commands. Metrics measured include the sum of wheel revolutions across all 6 wheels, and the sum of revolutions across all 4 steer actuators.

The first case is a typical approach to a waypoint during a drive. The nominal sequencing strategy is to use a GO_TO to get within 0.75-1.75 meters of a target, turn to face the goal if needed, and then perform a series of straight line motions to get right on top of it. Execution of the TURN_TO is controlled by a deadband parameter, such that a turn-in-place-maneuver will only happen if the rover is pointed sufficiently far from the goal at the end of the GO_TO leg. This is most



(a) Traditional method (b) New method

Figure 5: Simulated rover wheel tracks for a precision to a waypoint following a GO_TO that terminated with 20 cm of positional error. The sharp changes in the orange lines in the left-side Traditional motion indicate a turn-in-place maneuver, which results in signficant steering motor usage.

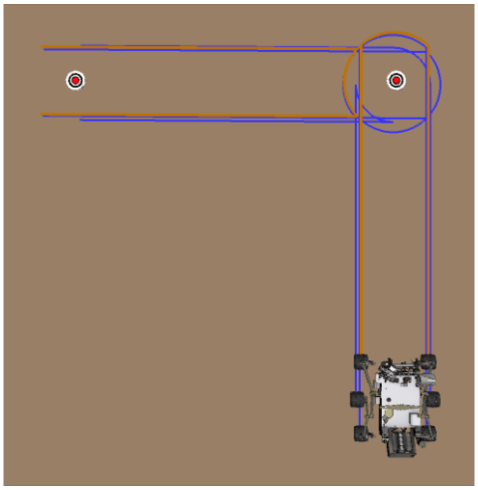
Table 1: Comparison of drive and steer actuator revolutions using traditional sequencing methods and modified method with the new commands.

	Wheel Revolutions [deg]	**Steer Revolutions [deg]**
Traditional (a)	5077.2	373.8
Arcs (b)	4833.3	33.1
% Difference	-4.8%	-91.2%

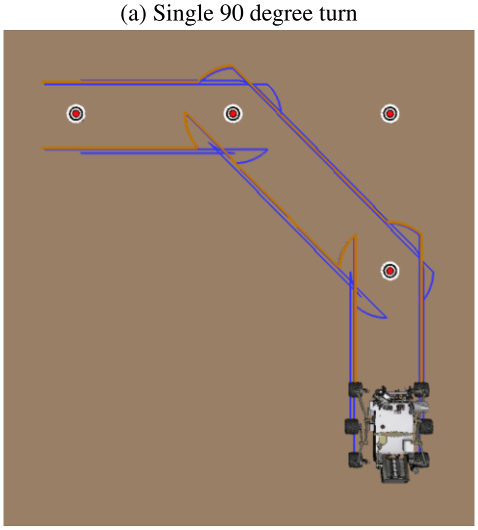
likely to happen when the rover is traversing high slip terrain like sand or steep slopes where turns would otherwise be preferentially avoided due to concerns with drive accuracy and wheel wear. The two frames in Figure 5 show what the wheel tracks look like for (a) the traditional sequencing method and (b) an updated method using the new arcing commands. A comparison of actuator usage for this case is given in Table 1, which demonstrates a small improvement in wheel driving rotations, and an order of magnitude improvement in steering motor usage.

The second case demonstrates navigation around an obstacle. Typical sequencing would include a series of straight line motions with turn-in-place maneuvers between them. This significantly increases total steer actuator motion during a drive compared to a similarly executed arcing maneuver. The three frames in Figure 6 show what the wheel tracks look like for (a) the traditional sequencing method using a single 90 degree turn, (b) the traditional sequencing method using two 45 degree turns, and (c) an updated method using the new arcing commands. A comparison of actuator usage for this case is given in Table 2. As in the prior case, we see some improvement in wheel revolutions, and a much larger improvement in the reductions of steering motor usage.

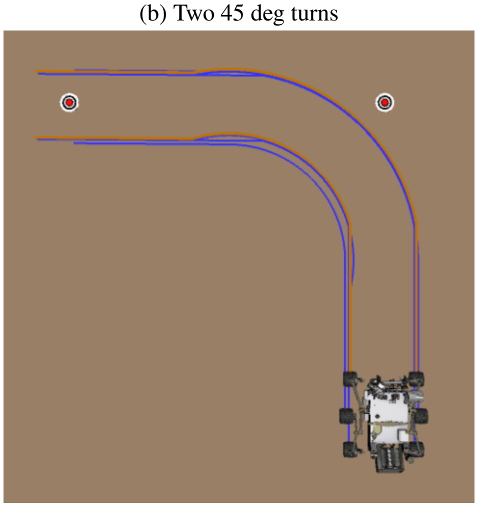
The data for these two scenarios show that although the



(a) Single 90 degree turn



(b) Two 45 deg turns



(c) Arcing

Figure 6: Simulated rover wheel tracks for a situation where the rover must complete a bootleg around an obstacle.

Table 2: Comparison of drive and steer actuator revolutions using traditional sequencing methods and modified method with the new commands for a dogleg motion.

	Wheel Revolutions [deg]	**Steer Revolutions [deg]**
Single turn (a)	30566.3	373.8
Double Turn (b)	26608.1	474.6
Arcs (c)	24582.7	111.5
% Difference a→c	-19.6%	-70.6%
% Difference b→c	-7.6%	-85.2%

reduction in drive actuator usage is modest, steer actuator usage can be reduced by upwards of 90% for certain maneuvers. This is largely because a turn-in-place requires the same large amount of steer actuator movement regardless of whether the desired change in heading is large or small. In contrast, the amount of steering motion required for arc commands primarily depends directly on how far the goal is off in heading, and inversely on the distance remaining to the goal.

4. The Math of ARC_TO

In this section, we derive the underlying mathematics for the new rover commands ARC_TO and ARC_UNTIL_TO.

ARC_TO commands the rover to perform an arc that will point the rover's front or rear at the navigation goal. The inputs to ARC_TO are the *goal position*, the *desired signed arc length*, and the *waydisc radius*. The *waydisc* is a disc around the navigation goal that the rover should not enter. The outputs are the change in heading, and if the rover would meet the waydisc, the shortened arc length.

Figure 7 illustrates the ARC_TO command and demonstrates the basic math used in this section.

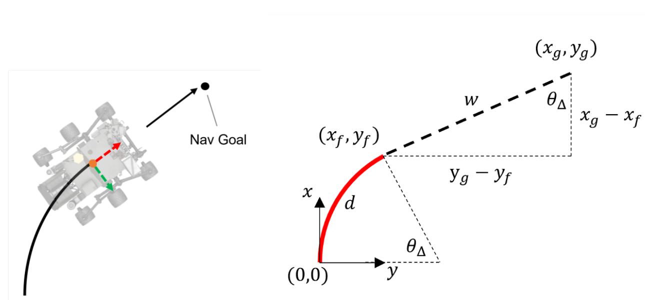


Figure 7: Rover points at navigation goal in ARC_TO command

If the rover meets the waydisc

First, we find the change in heading, θ_Δ and signed arc length d that will move the rover such that it simultaneously meets the waydisc and points at the navigation goal. In this case, the final position (x_f, y_f) is bound by the equation of an arc

(1):

$$
\begin{aligned}
x_f &= \frac{d}{\theta_\Delta}\sin\theta_\Delta = x_g - w\cos\theta_\Delta \\
y_f &= \frac{d}{\theta_\Delta}\sin\theta_\Delta = y_g - w\cos\theta_\Delta
\end{aligned} \tag{1}
$$

and by the waydisc radius and the heading at that point (2):

$$
\begin{aligned}
d =& \frac{\theta_\Delta}{\sin\theta_\Delta}(x_g - w\cos\theta_\Delta) \\
=& \frac{\theta_\Delta}{1-\cos\theta_\Delta}(y_g - w\sin\theta_\Delta)
\end{aligned} \tag{2}
$$

Each equation can be solved for d, the arc length. Then the arc length equations can be set equal and solved for θ_Δ, the change in heading (which can then be substituted back in):

$$\theta_\Delta = 2\arctan 2(y_g, x_g + w) \tag{3}$$

If the solution for arc length d is less than or equal to the commanded arc length, then a solution has been found where the rover meets the waydisc.

If the rover does not meet the waydisc:

If the solution for d is less than the commanded distance, the rover will not reach the waydisc (w unknown) and will not point at the goal. In this case, d will be the commanded arc length, and the change in heading θ_Δ must be found. Using the vector from the final position to the goal, we can describe the change in heading as an arc tangent, but this is also a function of the change in heading.

$$
\begin{aligned}
\theta_\Delta =& \arctan 2(y_g - y_f, x_g - x_f) \\
=& \arctan 2\left(y_g - \frac{d(1-\cos\theta_\Delta)}{\theta_\Delta}, x_g - \frac{d\sin\theta_\Delta}{\theta_\Delta}\right)
\end{aligned} \tag{4}
$$

We have not found a closed-form solution for θ_Δ, but we can use bisection search to find the zero-crossing of the equation. First, loop over values of $\theta_{\Delta i}$ until a reasonable interval is found. Then use bisection search to narrow the interval $[\theta_{\Delta i-1}, \theta_{\Delta i}]$ to a reasonably-accurate value.

5. THE MATH OF ARC_UNTIL_TO

ARC_UNTIL_TO is a flight software command which causes the rover to perform an arc of given length and change in heading until one of three ending conditions is met (see Figure 8). We would like to end the arc when pointed at the goal, but we would also like to prevent entering the waydisc and getting too close to the goal, or traveling away from the goal. We will find the fractions of the commanded arc that meet each of the conditions (if at all), and the smallest found fraction will be the fraction of the arc that is returned.

Rover pointing at goal

First, we find the fraction of the arc that will point at the goal. This is similar to the previous problem, but the radius of the arc is fixed, leading to (5) a closed-form solution for the relation between the goal and rover positions.

$$\tan k\theta_\Delta = \frac{y_g - y(k)}{x_g - x(k)} = \frac{y_g - r(1-\cos k\theta_\Delta)}{x_g - r\sin k\theta_\Delta} \tag{5}$$

Using trigonometric identities and the linear combination of sinusoids, we find (6) the solution for k.

$$
\begin{aligned}
k =& \frac{1}{\theta_\Delta}\Bigg[\pm \arccos\frac{r}{\sqrt{x_g^2 + (r-y_g)2}} \\
& + \arctan 2(x_g, r - y_g) + 2\pi n\Bigg]
\end{aligned} \tag{6}
$$

Several insights come from Equation 6. From left to right, the $\pm$ gives two solutions along the circle corresponding to the arc: one for pointing the front of the rover, and one for pointing the back. If the input to arccosine has magnitude > 1, there is no solution. This is the case if the goal is within the arc and the rover will never point at the goal. Similarly, if the input to arctan is invalid, the goal is at the center of the arc, and again there is no solution. The $2\pi n$ relates to the fact that the rover could theoretically travel in circles again and again, pointing at the goal each cycle. We must loop over values of n to find the smallest positive solutions for k.

Rover meeting waydisc

Figure 10 demonstrates the scenario where the rover touches the waydisc.

We find the fractions of the arc that intersect with the waydisc, using the Pythagorean theorem to relate the waydisc radius to the rover position (7).

$$
\begin{aligned}
\omega^2 =& \big(x(k) - x_g\big)^2 + \big(y(k) - y_g\big)^2 \\
=& \big([r\sin k\theta_\Delta] - x_g\big)^2 + \big([r(1-\cos k\theta_\Delta)] = y_g\big)^2
\end{aligned} \tag{7}
$$

$$
\begin{aligned}
k =& \frac{1}{\theta_\Delta}\Bigg[\pm \arccos\frac{2r(r-y_g) + x_g^2 + y_g^2 - \omega^2}{2r\sqrt{x_g^2 + (r-y_g)^2}} \\
& + \arctan 2(rx_g, r(r-y_g)) + 2\pi n\Bigg]
\end{aligned} \tag{8}
$$

The solution (8) for k also uses linear combination of sinusoids for similar reasons. The $\pm$ relates to how the arc can intersect with the waydisc twice. If the input to arccos has magnitude > 1, there is no solution because the arc will never intersect the waydisc. And lastly the $2\pi n$ shows how the rover can circle and intersect the waydisc again and again.

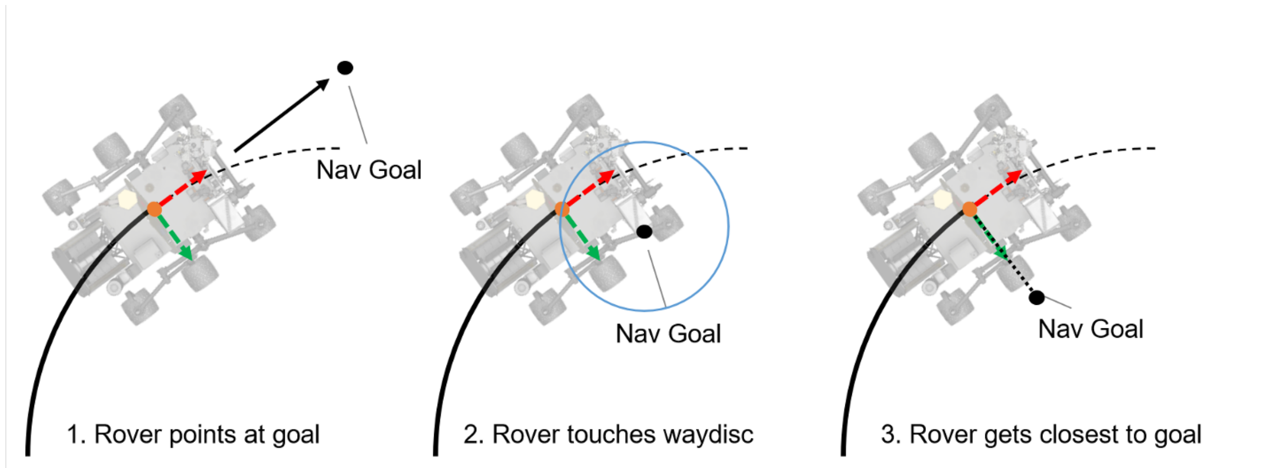


Figure 8: Three successful ending conditions for ARC_UNTIL_TO: (1) rover points at the navigation goal, (2) rover touches waydisc, and (3) rover gets closest to the goal line. Case 1 helps ensure that the rover is pointed at the goal, so that the next command will steer the wheels straight ahead. Case 2 gives Rover Planners the option of stopping once the rover is close enough to the goal such that no further motion is needed (any such motion might require very tight steering angles). And case 3 ensures we do not drive past the goal, even if the rover has yawed away from it due to unexpected slip. Cases 2 and 3 are inherited from the legacy ARC_UNTIL command.

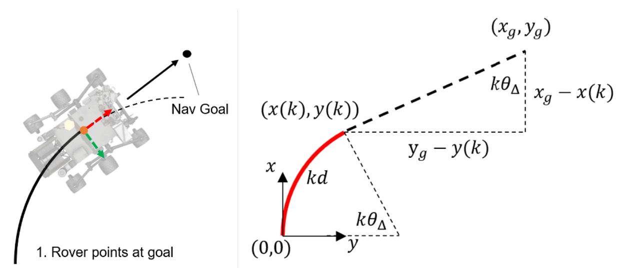


Figure 9: Solution space for ARC_UNTIL_TO when rover points at the navigation goal

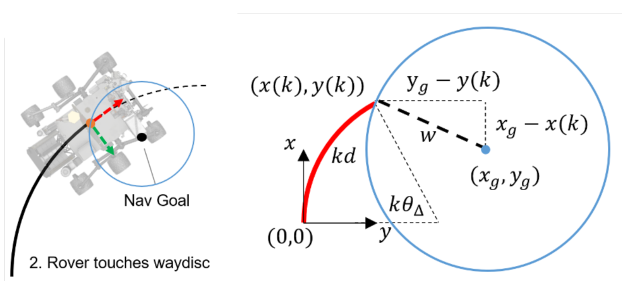


Figure 10: Solution space for ARC_UNTIL_TO where the rover touches the waydisc

Rover's closest approach to goal

Finally, we would like to prevent moving away from the goal by finding the position along the arc that is closest to the goal (Figure 11). We can show that the closest point of the arc is on the line defined by the goal position and the center of the arc which is at coordinates $(0, r)$. Use the tangent relationship to find the fraction of the arc that meets the matching angle. This formulation only fails if the goal is at the exact center of

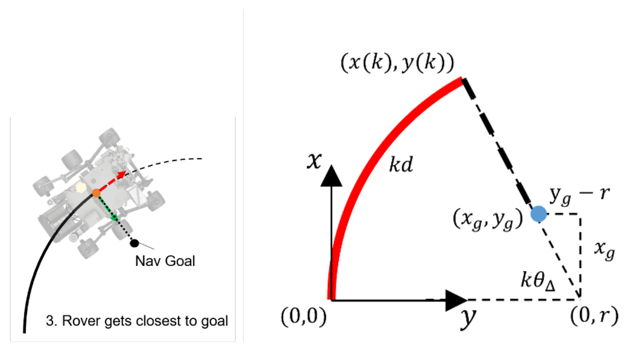


Figure 11: Solution space for ARC_UNTIL_TO to prevent the rover from moving away when it reaches closest approach to the goal

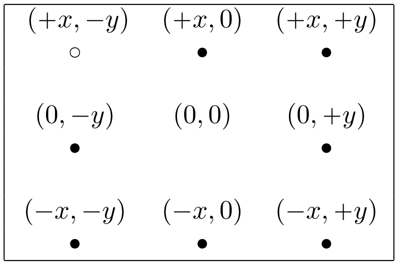
the arc (9).

$$\begin{aligned} \tan k\theta_\Delta &= \frac{y_g - r}{x_g} \\ k &= \frac{1}{\theta_\Delta}\left[\arctan 2(y_g - r, x_g) + 2\pi n\right] \end{aligned} \tag{9}$$

6. TESTING

Verification and Validation (V&V) Test Preparation

A thorough test plan was developed to verify and validate the new ARC_TO and ARC_UNTIL_TO prior to testing in JPL's Mars Yard. While the two commands are ostensibly similar their behavior can be quite different in comparable commanding scenarios, so two sets of tests were written to cover all cases. The test cases developed to verify and validate the new commands were designed to exercise nominal functions, as well as off-nominal usage. We wanted to show that the commands worked as intended, and also use them in



Legend:
- ● Motion expected
- ○ No motion expected
- Not tested

Figure 12: Goal locations visualization used in Table 3 and Table 4 gives gives a visual representation of tested goal locations in 8 possible areas relative to the rover: front, front-right, right, back-right, back, back-left, left, and front-left, with $+x$ corresponding to forward in the rover frame, and $+y$ corresponding to right in the rover frame

ways that were not intended and verify appropriate behavior. The V&V design flow started with identifying the test cases, describing the expected outcomes, checking the outcomes in simulations, then finally run for record on a flight-like testbed.

Test cases were chosen by simplifying all possible paths and positions into nine goals, a strategy that was employed by early mission testing of mobility commands. Eight basic directions such as straight forward or diagonally back and left, and one close to or at the rover origin reduced the tested combinations to a manageable workload. Goals were expressed in generic Cartesian coordinates, but the actual goal coordinates depended on the specific test case. The expected behaviors for the test cases were first described in the test matrix based on explanations of the commands provided by the developer.

Simulations of the test cases were then run to check the described behaviors. Several deviations were discovered during this process, necessitating large corrections to the test matrix but improving our understanding of the commands' behaviors. The simulations were invaluable in reducing the overall test time. Test cases could be run in a fraction of the time with accelerated drive speeds and quick resets. This allowed for rapid corrections and development of tests without the complexities of operating a real vehicle.

After correcting for the deviations discovered in simulated testing, the test scenarios were written for both commands with each scenario consisting of two to eleven individual tests. These scenarios are summarized in Tables 3 and 4, with ARC_TO consisting of 12 test scenarios, and ARC_UNTIL_TO having 13.

With the test cases written, a review was held where the test plan was presented to Mobility Subject Matter Experts, Rover Operations Team Chiefs, the R13 Flight Software Lead, and the Testbed Team for approval. The plan was approved with one open lien to add GUARDED Mode and Visual Odometry (VO) scenarios [5] to each of the two tests. Once these scenarios were added, the test campaign was ready to begin.

Testing in the Mars Yard

The new commands were tested using the Vehicle System Testbed (VSTB), a full engineering model of Curiosity with flight-like avionics and mobility system. Testing was complicated by the COVID-19 pandemic and other issues described in another paper about a different new mobility capability in the R13 flight software [9]. Initial testing was performed in the In Situ Instrument Lab (ISIL) in the same small test area described in the preceeding paper. But we were ultimately able to retest in a larger outdoor space.

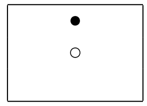
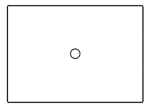
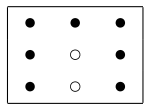
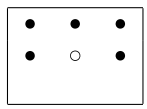
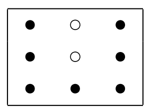
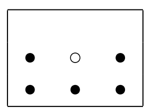
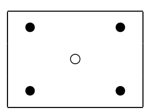
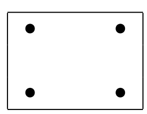
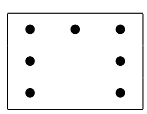
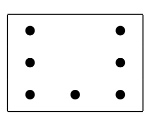
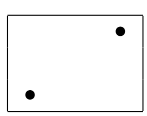
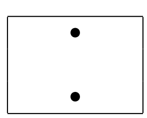
The VSTB was deployed on flight-like terrain in our outdoor Mars Yard test area to provide higher confidence for usage in flight compared to the purely simulated tests used to develop the test framework. With 25 different test scenarios and the MSL VSTB's top speed clocking in at a blistering 4.2 centimeters per second, V&V of these two commands spanned 5 separate days. A typical test shift required 3 team members, one to act as the Test Conductor, and the other two to act as Test Buddies. The Test Conductor was responsible for sending the commands specified by the test plan to the VSTB and recording any data necessary to verify that the vehicle's behavior matched what was expected. The two Test Buddies were responsible for moving the umbilical cable that provides both power and communication to and from the VSTB as it moves around the Mars Yard (it is too heavy for the rover to drag), using the panic button to stop vehicle motion if it is at risk of damage, as well as (if necessary) assisting the Test Conductor in understanding why rover behavior did not match what was expected by the test plan.

Test Shifts usually kicked off around 8 am, but due to the roughly one hour boot time of the VSTB, the Test Conductor would come in earlier to begin the power on procedure, later joined by the Test Buddies. After pulling out of the Mars Yard's garage, each test case would be commanded one after another with all team members working together to check that the rover drove in the expected way. On average, a shift covered 123 meters of driving, 5 times further than the average daily drive distance of the Curiosity rover on Mars [5]. With the shifts completed, the mobility team then presented their results to the same board that approved the test procedure. The board was satisfied with the Verification and Validation of these two new commands, and approved them for deployment to Mars as a part of the R13 flight software package.

7. CONCLUSION

The wheel wear issue first discovered on the Curiosity Mars Rover in 2013 is still a concern for future operations. We describe two new mobility commands, ARC_TO and ARC_UNTIL_TO, that are slated to be uplinked to the rover during its next major flight software upgrade in early 2023. These commands were developed as part of our long term response to these wheel wear concerns and will allow the operations team to precisely approach drive goals using minimal steering motion on hazardous terrain. The expected behaviors of the new commands and the mathematics behind them are described in this paper. The software successfully completed a thorough V&V process including simulating the commands in a virtual environment before deploying them on JPL's engineering testbed in the outdoor Mars yard. Expected command behaviors were exhibited and documented, and the new software has been officially approved for flight. As Curiosity enters its second decade on Mars, the ARC_TO and ARC_UNTIL_TO commands will help the operations team

Table 3: Overview of ARC_TO Test Scenarios. Goal locations under test shown in the rover frame (see Figure 12)

N	Test Scenarios	Goal Test	Scenario Behavior Verification
1	Fault Protection		Rover motion stops when fault conditions are met
2	Invalid Commanding		Commands with invalid arguments are rejected by FSW
3	Forward, Achieve heading		Motion is completed when rover is pointed at the goals facing forward
4	Forward, Enter waydisc		Motion is completed when Rover enters waydiscs facing forward
5	Backward, Achieve heading		Motion is completed when rover is pointed at the goals facing backward
6	Backward, Enter waydisc		Motion is completed when Rover enters waydiscs facing backward
7	Zero Distance, Nominal Heading Change		Rover turns in place to point at goals
8	Goal with no waydisc close to RNAV		Rover makes short-sharp arcs that end on top of the goals or pointing at them
9	Forward, longer distance than needed		Rover completes arcs at the goal in less distance than supplied
10	Backward, longer distance than needed		Rover completes arcs at the goal in less distance than supplied
11	Distance opposite goal		Rover completes arcs pointing at the goal
12	GUARDED mode and VO		Rover does not move when obstacles are detected

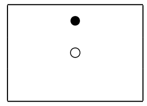
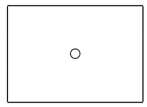
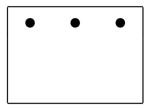
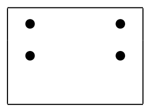
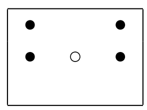
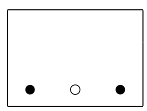
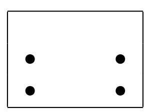
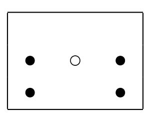
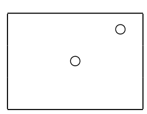
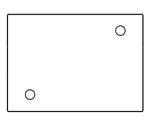
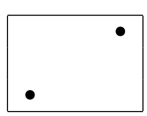
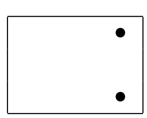
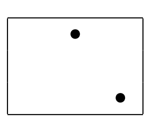
prolong the life of the vehicle and maximize the science return of the mission.

ACKNOWLEDGMENTS

The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration (80NM0018D0004). The authors would like to thank the Mars Science Laboratory Program for supporting this research.

Thanks to Jeff Biesiadecki for discussions about the earlier MER proposed commands.

Table 4: Overview of ARC_UNTIL_TO Test Scenarios. Goal locations under test shown in the rover frame (see Figure 12)

N	Test Scenarios	Goal Test	Scenario Behavior Verification
1	Fault Protection		Rover motion stops when fault conditions are met
2	Invalid Commanding		Commands with invalid arguments are rejected by FSW
3	Forward, Deadband		Rover does not move when a command within deadband is issued while facing forward
4	Forward, Achieve heading		Motion is completed when rover is pointed at the goals facing forward
5	Forward, Enter waydisc		Motion is completed when Rover enters waydiscs facing forward
6	Backward, Deadband		Rover does not move when a command within deadband is issued while facing backward
7	Backward, Achieve Heading		Motion is completed when rover is pointed at the goals facing backward
8	Backward, Enter waydisc		Motion is completed when Rover enters waydiscs facing backward
9	Zero Distance, Nominal Heading Change		No motion occurs
10	Distance opposite goal		Rover does not move because it would be moving further from the goal
11	Goal lines		Rover completes drives on the goal lines
12	Standoffs		Rover completes drives short of the goals, and does not start when within standoff
13	GUARDED mode and VO		Rover does not move when obstacles are detected

REFERENCES

[1] Olivier Toupet, Jeffrey Biesiadecki, Art Rankin, Amanda Steffy, Gareth Meirion-Griffith, Dan Levine, Max Schadegg, Mark Maimone, "Terrain-adaptive Wheel Speed Control on the Curiosity Mars Rover: Algorithm and Flight Results", Journal of Field Robotics, Volume 37, Issue 5, Special Issue on Space Robotics, August 2020, 699 - 728, 26 August 2019.

[2] R.E. Arvidson, P. DeGrosse, J.P. Grotzinger, M.C. Heverly, J. Shechet, S.J. Moreland, M.A. Newby, N. Stein, A.C. Steffy, F. Zhou, A.M. Zastrow, A.R. Vasavada, A.A. Fraeman, E.K. Stilly, "Relating geologic units and mobility system kinematics contributing to Curiosity wheel damage at Gale Crater, Mars", Journal of Terramechanics, Volume 73, October 2017, 73–93.

[3] Patrick DeGrosse, "MSL Wheel Tiger Team Final Summary Report: What is Causing the Damage?", internal presentation, September 2014.

[4] A. Rankin, N. Patel, E. Graser, J.-K. F. Wang, and K. Rink, "Assessing Mars Curiosity rover wheel damage", IEEE Aerospace Conference, Big Sky, Montana, USA, March 2022

[5] A. Rankin, M. Maimone, J. Biesiadecki, N. Patel, D. Levine, O. Toupet, "Mars Curiosity Rover Mobility Trends During the First Seven Years", *Journal of Field Robotics*, January 2021.

[6] N. Abcouwer, et al., "These Wheels Are Made For Arcing: Two New Mobility Commands to Reduce Wheel Wear", Flight Software Workshop, February 2022

[7] Jeffrey J. Biesiadecki, Mark W Maimone, "The Mars Exploration Rover Surface Mobility Flight Software: Driving Ambition," IEEE Aerospace conference proceedings, Big Sky, Montana, USA, 08 March 2006

[8] Alexandra Holloway, Jonathan Denison, Neel Patel, Mark Maimone, Arturo Rankin, "Six Years and 184 Tickets: The Vast Scope of the Mars Science Laboratory's Ultimate Flight Software Release", IEEE Aerospace Conference, Big Sky, Montana, USA, March 2023.

[9] M. Maimone, N. Patel, A. Sabel, A. Holloway, A. Rankin, "Visual Odometry Thinking While Driving for the Curiosity Mars Rover's Three-Year Test Campaign: Impact of Evolving Constraints on Verification and Validation", IEEE Aerospace Conference, Big Sky, Montana, USA, March 2022

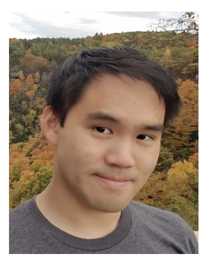
BIOGRAPHY

Mark Maimone *is a Robotic Systems Engineer in the Robotic Mobility group at the Jet Propulsion Laboratory. Mark designed and implemented the GESTALT self-driving surface navigation Flight Software for MER and MSL missions; during MSL operations served as Deputy Lead Rover Planner, Lead Mobility Rover Planner and Flight Software Lead; developed downlink automation tools for MER and MSL; and is now the Mars 2020 Robotic Operations Deputy Team Chief, and a member of the Rover Planner and Rover FSW development teams. He holds a Ph.D. in Computer Science from Carnegie Mellon University.*

Neil Abcouwer *Neil Abcouwer was a Robotics Systems Engineer at the Jet Propulsion Laboratory at the time of this research. He has worked on various robotic software projects, including mobility, pointing, pose estimation, and compression modules for the Perseverance Mars rover. He received his M.S. in Robotics from Carnegie Mellon University.*

PJ Rollins *is a Systems Engineer in the Engineering Operations for Surface Missions group at JPL. He is a member of the MSL Operations Team, where he supports the Mobility and Mechanisms Subsystem in both tactical downlink and long-term strategic roles. He received his B.S. in Aerospace Engineering from California Polytechnic State University, San Luis Obispo.*

Evan Hilgemann *is a Mechanical Engineer in the Technology Infusion group at JPL. He is involved in MSL operations as both a Rover Planner and a member of the Mobility/Mechanisms downlink team. In addition he has supported various flight and R&D projects as a hardware cognizant engineer. He earned an M.S. in Aerospace Engineering from the University of Michigan and a B.S. in Mechanical Engineering from the University of Nebraska.*



J. Freddy Wang *is a Mechatronics Engineer in the Mechanisms and Mobility group at JPL. He is currently a cognizant engineer for Mars Sample Return actuators. Previously he built and tested heritage actuators and seal dispensing mechanisms for the Mars 2020 mission, before joining MSL as a member of the Mobility/Mechanisms team. He received his Bachelor of Science in Mechanical Engineering from Rensselaer Polytechnic Institute.*

Nikunj Patel *is a V&V Lead for Lunar Permanence Mission Operations team for Blue Origin. Prior to that, he was one of the flight directors for MSL and the mobility subject matter expert for the rover. He achieved his Masters of Science in Aerospace Engineering from University of Central Florida (UCF) where he lead two cube satellite development missions through AIAA and UCF. In addition to that, Nikunj designed the first Ground Support Biobarrier (GSB) that would be utilized by NASA's Planetary Protection (PP) team to meet stringent bioburden requirements for future life detection missions to other terrestrial bodies.*

Alexandra Holloway *aims to design flight software for ground operability. She leads the flight software team for the Curiosity rover, chairs the data management subsystem in tactical operations, and serves as Assistant MSL Engineering Operations Team Chief. Prior to her work on MSL, Alexandra designed tools and processes for operators of the Deep Space Network as the system transitioned to Follow-the-Sun operations. Alexandra received her Ph.D. in Computer Science at the University of California, Santa Cruz, specializing in human-centered design and storage systems.*