Driving ATHLETE: Analysis of Operational Efficiency

Julie Townsend* and David Mittman†

Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91105, USA

The All-Terrain Hex-Limbed Extra-Terrestrial Explorer (ATHLETE) is a modular mobility and manipulation platform being developed to support NASA operations in a variety of missions, including exploration of planetary surfaces. The agile system consists of a symmetrical arrangement of six limbs, each with seven articulated degrees of freedom and a powered wheel. This design enables transport of bulky payloads over a wide range of terrain and is envisioned as a tool to mobilize habitats, power-generation equipment, and other supplies for long-range exploration and outpost construction. In 2010, ATHLETE traversed more than 80 km in field environments over eight weeks of testing, demonstrating that the concept is well suited to long-range travel. However, while ATHLETE is designed to travel at speeds of up to 5 kilometers per hour, the observed average traverse rate during field-testing rarely exceeded 1.5 kilometers per hour. This paper investigates sources of inefficiency in ATHLETE traverse operations and identifies targets for improvement of overall traverse rate.

I. Introduction

The All-Terrain, Hex-Limbed, Extra-Terrestrial Explorer (ATHLETE) is a multi-functional mobility and manipulation concept envisioned to support NASA activities in a variety of space environments. ATHLETE is a flexible robotic system consisting of a hexagonal platform supported by six articulated robotic limbs, each of which can terminate in a wheel for mobility on planetary surfaces or a variety of tools for operations in low-gravity environments.

When configured for surface mobility, with a wheel on the end of each articulated limb, ATHLETE can negotiate a wide range of planetary surfaces. On benign terrain, the wheels enable driving to efficiently cover long distances. When the surface is too soft, steep, or rough for driving, the limbs are used for walking, permitting extraction from embedding and mobility progress through areas impassable to most wheeled rovers.

To demonstrate the ATHLETE concept, the Jet Propulsion Laboratory (JPL) has designed and constructed several prototype vehicles, referred to as Software Development Models (SDM). The primary platform for traverse demonstrations is the second-generation ATHLETE prototype, built in 2009 and referred to as SDM-T12.[1,2]

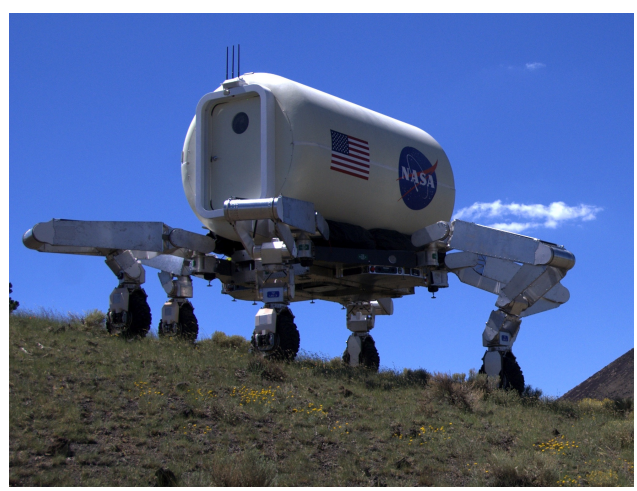

Figure 1. ATHLETE prototype carrying a μHab simulated cargo element during traverse testing at Black Point Lava Flow, AZ, September, 2010.

SDM-T12 consists of a pair of triangular three-limbed platforms called Tri-ATHLETEs which, when joined by a cargo pallet, form the hexagonal six-limbed system shown in figure 1. Sized to perform demonstrations at approximately one-half

*Robotics Software Engineer, Mobility and Manipulation Group, 4800 Oak Grove Drive, M/S 82-105, AIAA Senior Member.
†Supervisor, Planning and Execution Software Systems Group, 4800 Oak Grove Drive, M/S 301-250D.

lunar scale, it stands to a maximum height of just over 4 m and carries a payload of up to 450 kg in Earth gravity. Each limb has 7 degrees of freedom (DOF), six for precise positioning and one redundant pitch actuator to enable each limb to stow compactly.

Over the course of eight weeks in August and September 2010, the long-range traverse capabilities of the ATHLETE prototype were demonstrated. ATHLETE traversed over 80 km through desert and lava flow terrain, meeting and exceeding traverse milestones. However, analysis of traverse data showed that the average traverse rate was significantly slower than ATHLETE's typical ground speed during traverse, suggesting inefficiencies in traverse operations.[3] The sections that follow investigate the sources of these inefficiencies and identify potential remedies to improve traverse rates.

II. Traverse Operations

While ATHLETE's traverse performance at Black Point Lava Flow (BPLF) exceeded the milestone targets for both overall distance and distance traveled per day, analysis of traverse data revealed that ATHLETE's traverse capabilities were not exercised optimally. Average traverse rates for each day of testing ranged between 1 kph and 1.5 kph. Although this rate met the milestone requirement, it is remarkably slow when compared to ATHLETE's instantaneous ground speeds during traverse. Over 70% of the distance traveled by ATHLETE was traversed at speeds between 1.5 kph and 3 kph. This discrepancy suggests inefficiency in traverse operations. Further analysis revealed that ATHLETE stopped frequently during traverse, spending no less than 30% of traverse time, and typically more than 40%, sitting idle.[3]

In the field, ATHLETE traversed in one of two operational modes, either under control of a remote operator sitting at a workstation in the base camp or under control of a local operator walking alongside the vehicle.

A. Traversing under Remote Operator Control

When controlling ATHLETE from a remote location, the operator communicates with ATHLETE through a radio link, but has no direct visual contact with the vehicle. An operations safety officer accompanies the robot with a safety kill switch, but voice communication between the remote operator and the safety officer may or may not be present. The remote operator's workstation, shown in figure 2, may be located at the field site or another remote location and is composed of a general-purpose computer workstation with multiple displays used for tracking the performance of the vehicle.

During operation, the robot acquires stereo images of its environment and those images are displayed to the remote operator as an aid in determining the distance to obstacles and general environmental conditions. Remote operators drive the vehicle through the direct input of textual commands or through a variety of directional input devices such as joysticks. In response to predictions of improved vehicle reliability and performance in 2010, the operations team developed a simplified driving joystick based upon the Nintendo Wii Nunchuk Controller, which can be operated with one hand. The Controller features a joystick for directional control, roll-axis control for turning and a dead-man style kill switch.

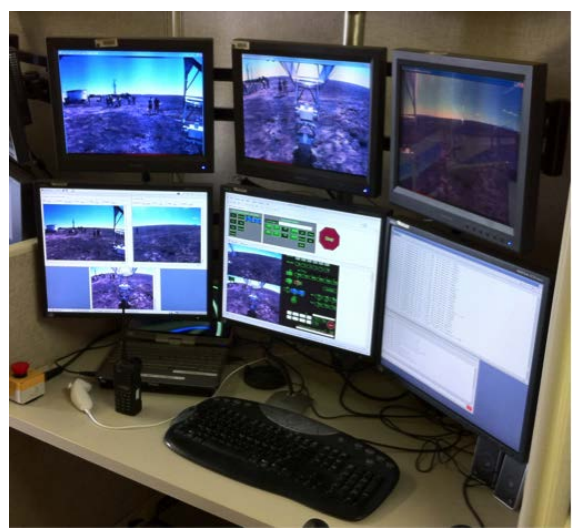

Figure 2. The Operator Control Workstation provides the displays and controls required to remotely operate the ATHLETE rover.

Because of the long traverses planned during the 2010 field test, the operations team was not guaranteed to have a working radio link between their field-deployable operations facility (a converted 40-foot cargo container), or OpsBox, and the vehicle. Safe vehicle operations require a low-latency link between the vehicle and its ground-based command and monitoring computer, which prior to 2010 were solely located within the OpsBox during field-testing. To accommodate this change in operating paradigm, the operations team developed a portable version of the ground-based command and monitoring

computer that can be mounted on the vehicle itself.

The portable operations workstation, or PortOps, shown in figure 3, was developed in two parts. A vehicle-mounted Linux laptop provided nearly all the functionality of the ground-based operations computer, and a handheld version connected wirelessly to provide limited commanding and telemetry monitoring capability to an operator walking alongside ATHLETE. This arrangement provided the reliable communications and operator interface required for safety and data logging. In addition, the PortOps handheld supported Wii Nunchuk driving, enabling the local operator to drive ATHLETE through areas with poor communications links to the remote operator's groundstation.

Figure 3. The PortOps Laptop and Handheld provide a complete command interface for a local operator, independent of the remote groundstation.

A remote operator could be expected to have good strategic situational awareness with plenty of processing power and screen room to display global mapping with robot location updated in real time from vehicle data. On earth this is GPS data. Local situational awareness is expected to be the difficulty for a remote operator and is managed as well as possible by good sensing and imagery from the vehicle supported by good ground processing and data visualization at the workstation.

In practice for ATHLETE at BPLF, GPS drift gave problematic localization of the vehicle and made it very difficult for remote operators in the OpsBox to determine if they were staying on course, particularly over short distances. As a result, strategic situational awareness for the remote operators was poor. Remote operators were frequently unaware of course deviations until informed by a local operator.

Local situational awareness for the remote operators was an even bigger problem than anticipated. ATHLETE's navigation cameras often returned images with insufficient contrast for detecting obstacles, particularly when traversing dark lava rock. The ground operations system was not designed to post-process the tactical images to improve contrast. In addition traversable obstacles like grasses and desert scrub in the images often obscured dangerous obstacles, like large rocks or crevices, making it impossible for remote operators to safely chart paths based on imagery alone.

In most mission concepts, the primary driver for an ATHLETE vehicle is a remote operator on Earth or in an orbiting spacecraft. In field-testing of the ATHLETE prototype, the goal was to demonstrate this capability and use it for the majority of traverse operations. Remote operation of ATHLETE in the field requires both data transfer from the robot to the remote groundstation and voice communication between the local and remote operators. Due to infrastructure limitations at Black Point Lava Flow, these requirements could only be met within 2 km of the Base Camp. For this reason, only 5 km of the 60 km traversed at Black Point Lava Flow (BPLF) was commanded remotely, as shown in figure 4. In addition, when remote commanding was attempted, infrastructure problems typically resulted in traverse pauses of such long duration that they were excuded from the previously published performance analysis.[3]

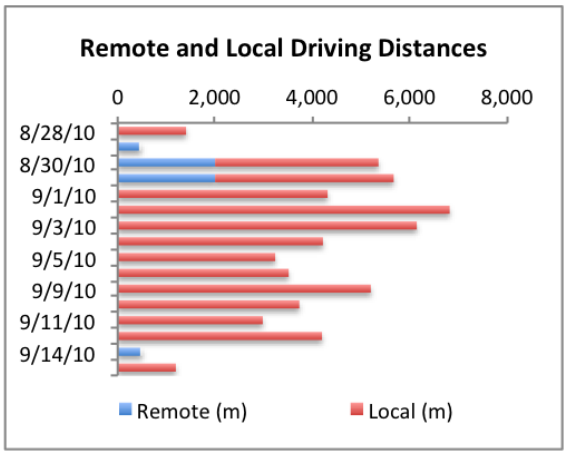

Figure 4. Remote operators controlled ATHLETE for a small portion of the total distance traveled.

B. Traversing under Local Operator Control

The local operator was intended to be an eyes-on-the-ground safety check for remote driving operations and a stopgap operator to move the vehicle through areas

with limited communications coverage. While a local operator could conceivably be an astronaut working alongside ATHLETE, the more likely scenario for long-range traverse is remote operations of ATHLETE while the astronauts interactively drive a nimble exploration vehicle. However, due to the aforementioned problems with remote operations, most of the traverse distance was covered using local operation.

A local operator walks alongside ATHLETE and controls the vehicle using the PortOps handheld, using the joystick to start and stop traverse, selecting drive direction via joystick position and path curvature via build-in tilt sensors within the joystick unit. A limited set of supporting commands is accessible through the touch screen on the handheld computer, controlling ground speed and general driving mode. In contingency situations, the local operator can access the full ATHLETE command set through the PortOps laptop.

A local operator is expected to have excellent situational awareness because he or she can see and interpret the terrain in ATHLETE's immediate vicinity. The local operator also has instant visual feedback on ATHLETE's behavior during the execution of each command, enabling constant evaluation of robot trajectory and quick corrections if necessary. Because the local operator is embedded with the robot, awareness of the current position on the strategic path requires extra information. In this case, the local operators were provided with a handheld GPS unit pre-programmed with waypoints along ATHLETE's intended path, giving them continuously updating direction and distance information.

Because of the difficulties with remote operation during the traverse testing at Black Point Lava Flow, the published performance statistics in general reflect performance during local operations.[3]

III. Analysis of Inefficiency

As mentioned in the previous section, ATHLETE's traverse rate in the field was significantly slower than its ground speed, indicating a lack of continuous driving. Analysis of traverse data reveals that traverse progress was frequently interrupted, and that these interruptions came from a variety of sources originating in both the ATHLETE system and the operator interface. Figure 5 illustrates the cause of termination of each drive command issued during traverse days at Black Point Lava Flow. In general, drive commands not followed by a new command within 10 minutes are excluded from this analysis, to avoid including lunch breaks and maintenance activities in the results. An exception was made for drive commands ending in a stall or motor controller error, to acknowledge that these errors often require more than 10 minutes to resolve.

As figure 5 shows, the vast majority of commanded drives ended in some off-nominal condition that stopped the vehicle and required initiation of a new command. The errors originate from idiosyncrasies distributed throughout the operational system, including ATHLETE hardware, ATHLETE software, and the PortOps controller.

Some commanding difficulties are unique to remote operations and since remote operation was infrequent at Black Point Lava Flow, figure 5 does not accurately represent their potential effects on long-range traverse operations. Remote operation primarily affects driving efficiency by increasing the time spent paused between mobility commands while the remote operator attempts to determine a safe and effective course of action. Remote driving can also decrease the population of medium- and long-duration mobility commands as the remote operator stops more often to assess the safety and trafficability of the terrain.

For each source of traverse inefficiency, measures can be taken to reduce or even eliminate the effect on ATHLETE's long-range traverse performance. The subsequent sections discuss each source of inefficiency in detail, examining its underlying cause, its overall effect on driving efficiency, and measures that may be taken to resolve the issue.

A. Heading Recommand and Jitter

The chart in figure 5 reveals that heading recommand and jitter accounted for more than half of all drive terminations. Drive commands included in this set traveled less than one meter before being interrupted by a new command or stopped by the operator. We attribute these very short drives to the difficulties of operating using PortOps, in particular the Wii Nunchuk.

The Nunchuk driving methodology was designed to accommodate the most common driving commands, including straight-line drive commands, arcing drive commands and commands to turn in place. The rolling motion of the major axis of the Nunchuk translated into a vehicle heading change, and positioning of the joystick translated to the initial departure direction. Operators found both of these controls difficult to

finesse, resulting in a trial-and-error approach to initiating each drive command. Local operators would start, stop, and restart motion until the controller produced the desired heading and arc radius. Obtaining the proper sensitivity of the heading change control was difficult because operators preferences for sensitivity varied widely. In addition, each Nunchuk device differed from another in the calibration of the raw data numbers generated by the joystick and the accelerometer. A manual calibration procedure was developed to accommodate this variation, but operators sometimes failed to execute the calibration. Even when the calibration had been performed, the Nunchuk sometimes lost calibration during operation.

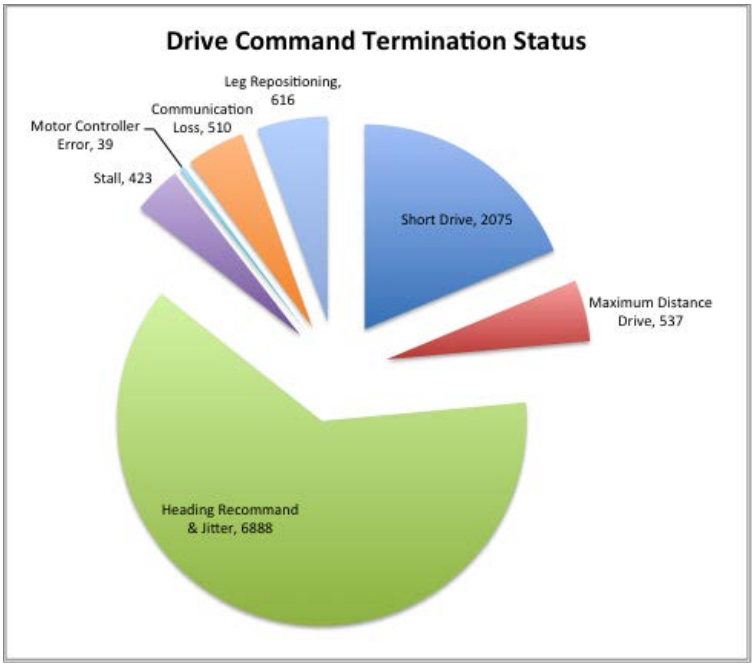

Figure 5. Early termination of ATHLETE drive commands caused frequent stops, resulting in inefficient traverse operations.

Jitter occurred when the Nunchuk issued new drive commands that superseded a drive already in progress, causing the vehicle to stop and re-steer. The Nunchuk was overly sensitive to the turning and arcing elements of the drives and often interpreted an operator's unsteady hand as a change in heading and a new command. To remedy this source of inadvertent commanding, an axis lock mechanism was created. By pressing and releasing a button on the Nunchuk, the operator could lock out the creation of additional drive commands. The use of the axis lock button, in conjunction with the dead-man button, required some difficult dexterous use of adjacent fingers on one hand, or the use of fingers on both hands. Releasing the dead-man switch while attempting to press the axis lock button was another source of inadvertent drive stops.

The data shows that improving the command interface can substantially improve driving efficiency under local operator control. A wide variety of solutions are under consideration. One possible approach maintains the current operations interface and focuses on improving the performance of the Nunchuk controller. Adding automatic calibration and fine-tuning the performance of the joystick and accelerometer inputs could reduce or eliminate the observed errors. Another option explores new operational philosophies, enabling the operator to direct ATHLETE toward a distant goal rather than manually specifying the wheel position and path curvature.

B. Stalled Joints

ATHLETE's control software includes a current limit for each actuator to prevent overheating of the motor if the joint is stalled. As shown in figure 5, this stall protection stopped more than 400 drive attempts at Black Point Lava Flow. The stalls had multiple sources, the most prominent of which were stalls in the steering actuators and stalls in the thigh pitch actuators. The contribution of each of these particular stall errors to the total number of stall errors is shown in figure 6.

1. Stalled Steering Actuators

As figure 6 illustrates, stalls of the steering actuators were a common occurrence, making up 64% of the errors due to stalled actuators. The reason these errors were so common is that they occurred in reaction to an idiosyncrasy of normal ATHLETE traverse operations.

The stall check is embedded in the firmware of the motor controller and controlled by two software variables, *climit* and *cmax*. *climit* represents the maximum average current allowable for the actuator, and *cmax* represents the maximum current the actuator can ever be permitted to receive. The motor controller

will not send more than *cmax* current to the actuator at any time, but will allow the actuator to receive transient current spikes above *climit* for up to three seconds before declaring a stall error.

Due to the high ground pressure on ATHLETE's tires, the steering actuators require a continuous current level greater than *climit* to steer the wheels when the vehicle is fully loaded for traverse. The actual current required is greater than *climit*, but nowhere near *cmax*. Because the maximum steering rate is approximately 28 degrees per second, steering changes of over 90 degrees require more than 3 seconds to complete, causing the motor controller to declare a stall in the course of a nominal steering activity.

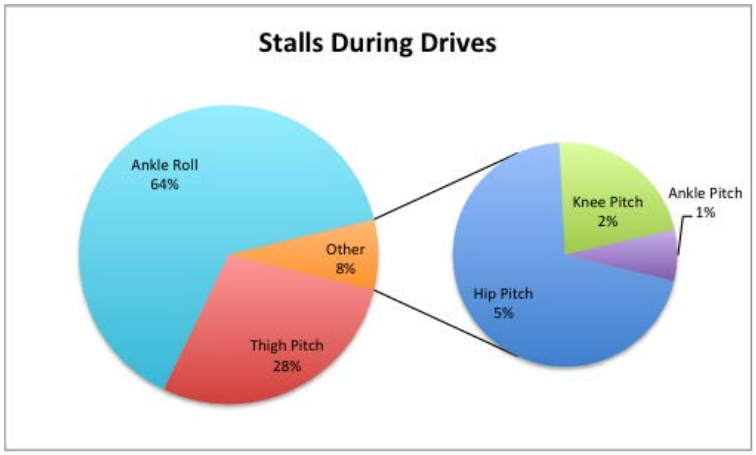

Figure 6. The ankle roll (steering) and thigh pitch actuators were the cause of the majority of stalls during drives at Black Point Lava Flow.

This situation represents a fundamental mismatch between the stall monitor and the nominal operating conditions, which should be addressed. While the three-second time horizon on the motor controller cannot be changed, it may be possible to avoid the stall errors by raising the *climit* for the steering actuators, if this is deemed safe by the mechanism experts, or by reducing the maximum steering rate to keep the actuators within the *climit* under nominal conditions. Eliminating unnecessary steering stalls could improve overall traverse rate by almost 2.5%.

2. Stalled Pitch Actuators

Another significant source of stalled joint errors are the limb pitch joints, in particular the thigh pitch joints which are responsible for 28% of all stall errors, and the hip pitch joints, which are responsible for 5% of all stall errors, as shown in figure 6.

Unlike the stalls on the steering actuators, the pitch joints typically stall under high load, with current levels very close to *cmax*. This indicates that these stalls reveal true high loads at the pitch joints. Early in testing, it was observed that pitch joint stalls were frequently attributable to attempts to reposition or reorient limbs while vehicle loading was unevenly distributed amongst the wheels.

While the onboard algorithm for active terrain compliance effectively distributes loads amongst the wheels during driving,[4] some common cases were identified in which the drive behavior attempted to reorient wheels while forces were not optimally distributed. For example, the drive behavior attempted to reorient all wheels before initiating each traverse command. If the previous drive activity ended with uneven wheel loading due to an error or other discontinuity, this pre-drive orientation would be performed with an uneven loading distribution, frequently resulting in a stall. In another case, repeated use of the active compliance behavior resulted in a large number of internal limb and body motions, with a side effect of wheels wandering away from their optimal kinematic drive poses. This also frequently resulted in stalled joints. Both of these issues were resolved in the early stages of traverse testing, the first by delaying wheel reorientation

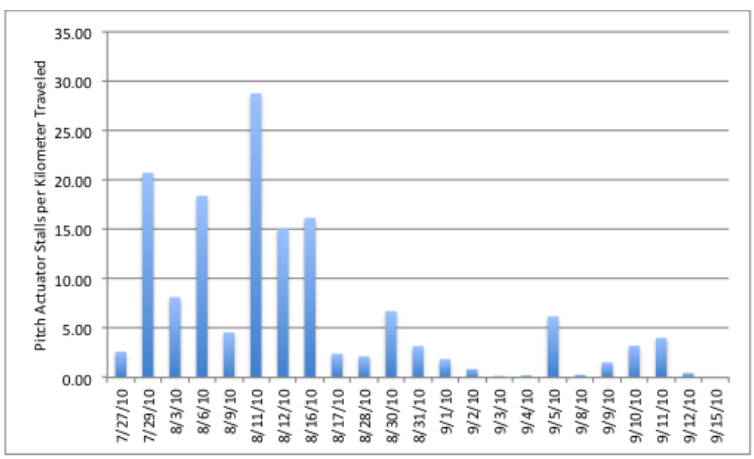

Figure 7. Pitch actuator stall rates decreased after changes to vehicle pose strategies were implemented.

until after the drive had begun and the wheel loading distribution was under active control, and the second by monitoring and correcting wheel positions autonomously.

Figure 7 shows the number of pitch joint errors on each test day. Tests conducted before the drive behavior upgrades had more stalls than later test days, after implementation of the changes. The changes to active compliance were effective in reducing pitch errors from an average of 15.44 stalls per kilometer traveled before BPLF to 2.26 stalls per kilometer traveled during BPLF for an effective reduction of 85%. At this point, there are probably no additional measures to be taken short of redesigning the joints to handle greater load or reducing the cargo weight of the prototype.

C. Motor Controller Errors

The SDM-T12 ATHLETE prototype has been plagued during operations with motor controller errors that resulted in frequent and sometimes long stops, occasionally requiring motors or controllers to be replaced. Investigation into this issue in preparation for traverse testing revealed that these errors most frequently resulted from erroneous readings of the virtual hall sensors built into each motor's encoder disk. Just before the traverse demonstration at BPLF, a solution was discovered and implemented in the motor controller firmware.

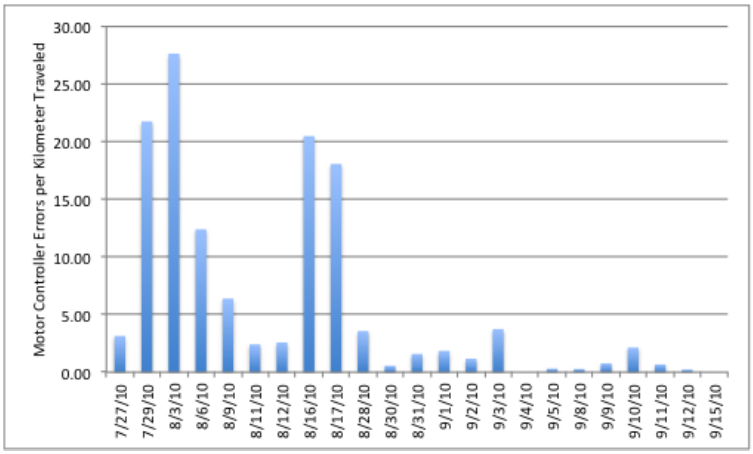

Figure 8. Motor controller errors per kilometer decreased after changes to controller parameters were implemented.

Figure 8 illustrates the improvement in drive performance following the motor controller firmware upgrade. The incidence of motor controller errors was drastically reduced, from dozens of errors per kilometer during early testing to consistently fewer than five errors per kilometer at BPLF. At Black Point Lava Flow, the effect of motor controller errors on drive efficiency was insignificant, accounting for less than 0.5% of all stops as shown in figure 5.

D. Communication Loss

A critical safety feature of the ATHLETE system is the command heartbeat, which verifies that the vehicle is under continuous operator control. If the PortOps laptop fails to receive a heartbeat signal from the PortOps handheld within a preset interval, the laptop concludes that operator control has been lost, and commands ATHLETE to halt all motion.

Figure 5 shows that communication loss between the PortOps laptop and handheld accounted for over 500 halts in traverse progress. Some small percentage of these halts were valid safety reactions caused by depletion of the handheld battery or the operator wandering too far away from the vehicle. Most, however, were due to performance issues on the PortOps handheld or laptop.

Occasionally, the PortOps handheld software would freeze or crash, causing a loss of communication. More frequently, processing delays would occur when the handheld or laptop were performing CPU-intensive operations, delaying the handling of the heartbeat signal and falsely declaring a communication loss. Better process management on both the laptop and the handheld would eliminate this source of inefficiency.

E. Wheel Repositioning

To solve problems with stalling pitch actuators, the ATHLETE drive behavior was upgraded to autonomously monitor and correct wheel positions. The implementation of this upgrade checks the wheel positions at the beginning of each drive command and, if necessary rolls up to three wheels at a time along the ground to the nominal driving pose. While this functionality is a great improvement in efficiency over recovery from stalled actuators, analysis shows that the repositioning activity itself was a source of significant lost time.

Over two hours of traverse time was spent on repositioning of limbs, accounting for approximately 4% of the total traverse time.

Large portions of the delays seen during limb repositioning were due to a problem with the behavior that rolls the wheels into position. While wheels are rolling, motion toward the goal position frequently slows to a halt. The stop goes undetected by the software, which continues running but making no progress until detected by an operator, stopped, and restarted. Unsurprisingly, this procedure results in significant lost time. Unfortunately, the cause of this problem is not well understood, and may require significant time and effort to investigate and correct. Also, while correction of this issue would significantly improve the efficiency of limb repositioning, rolling wheels into position while the vehicle is stationary will continue to be a source of lost traverse time. Ideally, future upgrades to the drive behavior will actively control wheel position during motion, eliminating the need for pre-drive adjustments. Integrating this type of active control into an already complex behavior is expected to be difficult and time-consuming.

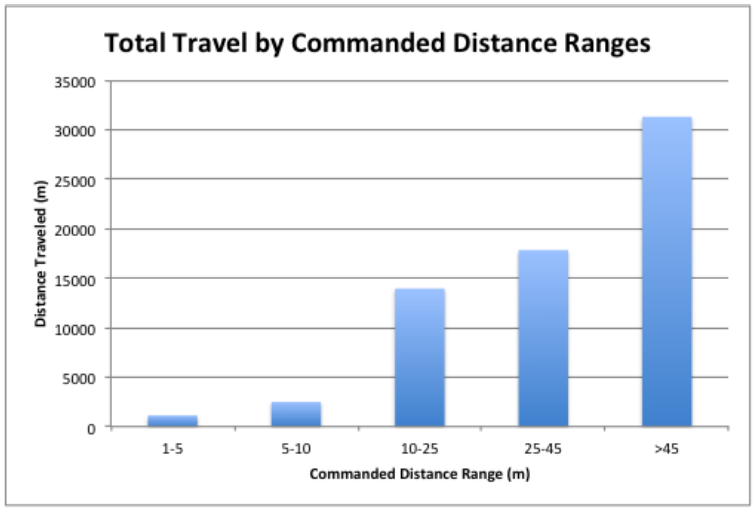

Figure 9. Length of ATHLETE traverse segments during BPLF testing

F. Drive Distance Limitations

Figure 9 illustrates the contribution of drive segments of different lengths to the overall traverse distance. Short drives of 1 to 10 meters were used when navigating difficult terrain, slopes, or the crowded conditions in the base camp. On benign, open terrain, ATHLETE traveled longer distances. Drives of 10 to 45 meters typically represent operator navigation, heading changes, and curving paths, while drives over 45 meters represent long, straight traverses.

Figure 10. Mobility command efficiency improves with increased maximum arc length by reducing the number of stops per kilometer.

One prominent source of inefficiency arose from the drive behavior implementation in ATHLETE's onboard software. A characteristic of this behavior limits commanded drives to no more than 50 meters in length. In the absence of difficult terrain and other system errors, ATHLETE paused at least 20 times per kilometer as a result of this distance limit. figure 5 shows that 537 pauses during traverse were due to the maximum drive distance limit.

While increasing this distance limit would improve drive performance by significantly reducing the number of stops per kilometer, as shown in figure 10, changing the command philosophy for long traverse operations could be even more effective. Particularly when operated by a remote operator, it is probably more efficient to set a waypoint in ATHLETE imagery and have the robot work out the direction and distance to the waypoint and break the drive up into its own manageable chunks. This paves the way for more autonomous onboard capability in obstacle avoidance and path planning. It also brings the operator up to a more strategic level in which an operator can evaluate a future strategic waypoint and update the vehicle's path seamlessly.

For local operations, if image analysis isnt handy, the vehicle could calculate an arbitrary waypoint in the correct direction to get the same effect. Changing to this type of operations concept would completely

eliminate inefficiencies due to distance limits.

IV. Conclusion

Analysis of data from long-range traverse testing of the ATHLETE prototype shows that traverse operations were inefficient and that these inefficiencies were due to frequent interruptions in traverse progress from a variety of sources. Examination of individual sources of traverse interruptions revealed that ATHLETE's average traverse rates could be significantly improved by addressing each of these issues.

Solutions to improve efficiency vary widely. Some, like revision of the ankle pitch current limit, have the potential for significant efficiency in return for relatively little effort. Others, including changes to the traverse philosophy and onboard software implementations have the potential to dramatically improve performance, but require significant time and manpower to realize.

Acknowledgments

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. ATHLETE development at JPL is lead by Principal Investigator Brian Wilcox and is conducted under the Human-Robot Systems Project led by Rob Ambrose of JSC with funding from the NASA Office of the Chief Technologist, Game Changing Division.

References

[1]Wilcox, B.H., ATHLETE: A Cargo-Handling Vehicle for Solar System Exploration, IEEE Aerospace Conference, 2011.

[2]Heverly, M., Matthews, J., Frost, M., and McQuin, C., Development of the Tri-ATHLETE Lunar Vehicle Prototype, Proceedings of the 40th Aerospace Mechanisms Symposium, May 2010.

[3]Townsend, J., ATHLETE Mobility Performance in Long-Range Traverse," AIAA Space 2011 Conference and Exposition, September 2011.

[4]Townsend, J., Biesiadecki, J., and Collins, C., ATHLETE Mobility Performance with Active Terrain Compliance, IEEE Aerospace Conference, 2010.