

# Measuring Robot Performance in Real-time for NASA Robotic Reconnaissance Operations

Debra Schreckenghost

TRAC Labs, Inc  
1012 Hercules, Houston, TX 77058  
ghost@ieee.org

Terrence Fong

NASA Ames Research Center  
Moffett Field, CA 94035  
terry.fong@nasa.gov

Tod Milam

TRAC Labs, Inc  
1012 Hercules, Houston, TX 77058  
tmilam@traclabs.com

Hans Utz

Research Institute for Advanced  
Computer Science  
Moffett Field, CA 94035  
hans.utz@nasa.gov

## ABSTRACT

Technical advances since Apollo make it possible to perform robotic reconnaissance to gain a better understanding of lunar sites prior to human exploration. NASA is conducting analog field tests to investigate these operations concepts with advanced robots and simulated flight operations. We have developed robot performance monitoring software for use during robotic reconnaissance operations. We measure robot performance by monitoring robot data in real-time and computing robot performance metrics from that data. Metrics are computed for two regimes of flight operations – remote supervision of autonomous robot operations and debrief support after a flight operations shift. In this paper we describe our performance monitoring software, define the metrics we compute, discuss how these metrics are used in flight operations, and summarize results from recent field tests.

## Categories and Subject Descriptors

C.4.3 [Performance of Systems]: Measurement techniques, Performance attributes, Reliability, availability, and serviceability; I.2.9 [Artificial Intelligence]: Robotics - Commercial robots and applications, Operator interfaces

## General Terms

Algorithms, Measurement, Performance, Experimentation

## Keywords

Operational performance metrics, robotic recon, space robotics.

## 1. INTRODUCTION

NASA's plan to return humans to the moon raises questions about how to conduct lunar exploration missions to best utilize limited crew resources. Technical advances since Apollo enable the use of robotic systems to complement the human mission. Improved instrumentation makes it possible to perform robotic reconnaissance, or "recon", to gain a better understanding of lunar sites prior to human exploration. NASA is conducting analog field tests to investigate these operations concepts with advanced robots and simulated flight operations [1].

We define robotic recon as operating a planetary rover under ground, or non-EVA astronaut, control to scout planned sorties

prior to EVA activity. Scouting is an essential phase of field work, particularly for geology. Robot instruments provide measurements of the surface and subsurface at resolutions and from viewpoints not achievable from orbit. This surface-level data can then be used to select locations for field work and prioritize targets to improve crew productivity. Robotic recon can be done months in advance, or be part of a continuing planning process during human missions. [1].

We first began studying robotic recon during the June 2008 NASA Human-Robotic System (HRS) project analog field test conducted at the Moses Lake Sand Dunes, WA. During this test, an experimental ground control team located at the NASA Johnson Space Center (JSC) used a K10 planetary robot (Figure 1) to remotely scout a portion of the sand dunes. The data collected during the reconnaissance was then used to develop a plan for crew EVA in the same area. Lessons learned at Moses Lake were subsequently used to improve and validate robotic recon systems during Operational Readiness Tests (ORT) at the NASA Ames Research Center (ARC) in November 2008 and June 2009. Most recently, we conducted an experimental assessment of robotic recon as part of the 2009 Desert Research and Technology Studies (D-RATS) analog field campaign at Black Point Lava Flow, AZ.

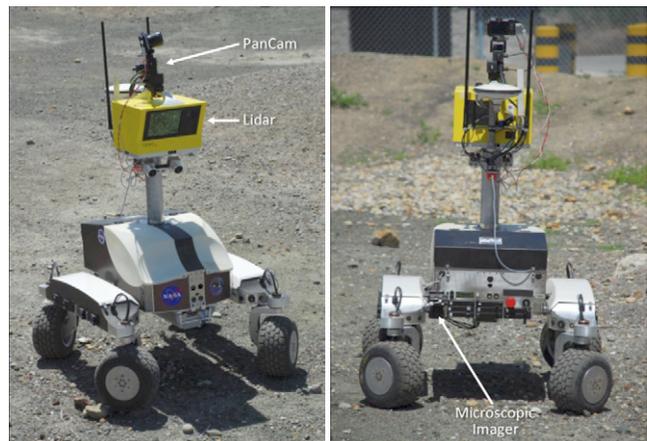
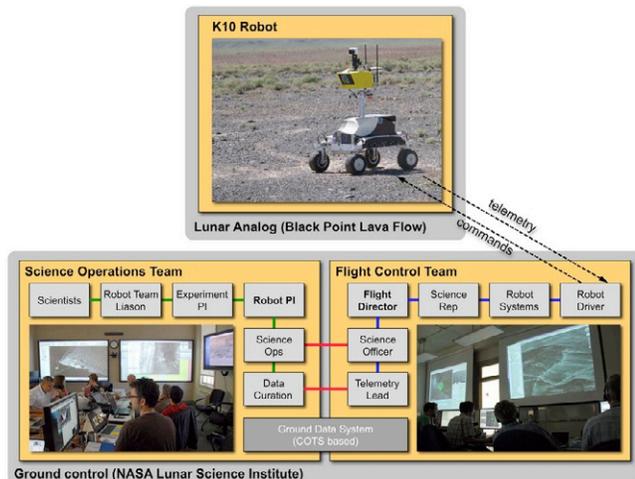


Figure 1. The K10 planetary rover is equipped with three instruments for robotic recon: a panoramic imager (PanCam), a 3D scanning lidar, and a terrain facing microscopic imager.



**Figure 2. An experimental ground control for robotic recon includes a science operations team and a flight control team. The roles and protocols used by this ground control are a hybrid of Apollo, Shuttle, Space Station, and MER concepts.**

For these ORTs and field tests, ground control (Figure 2) was conducted concurrent with robot surface operations. The Science Team builds robot plans to visit features of interest and take instrument readings to determine what features would benefit most from astronaut Extra-Vehicular Activity (EVA). During robotic recon operations, the ground control team reviews robot plans, then uplinks them to the robot. Similar to planetary rovers like JPL's Mars Exploration Rover (MER), the K10 robot immediately begins to execute the uplinked plan. But, unlike MER, ground control supervises the robot as it executes the plan. They continuously monitor robot data in real-time and can intervene using tele-operations if opportunities or problems arise.

To help improve the efficiency and effectiveness of robotic recon operations, we have developed real-time robot performance monitoring software. In the following, we describe our approach, define the performance metrics we compute, discuss how these metrics are used in robot operations, and summarize results from recent field tests.

## 2. RELATED WORK

The planetary surface environments in which recon robots must operate are highly variable. To maximize the value of the reconnaissance, the robot may be required to make observations in areas that are difficult to access or traverse. For example during the June 2009 ORT at NASA Ames Research Center, the K10 robot operated near its safety limits to access desired geologic features on steep slopes. When communication latencies and bandwidth permit, tele-operating the robot in these conditions can be preferable to autonomous operations. Thus, robotic recon includes autonomous robotic activities interleaved with scheduled tele-operation.

For this operations model, human-robot interaction metrics such as interaction efficiency [2] and neglect tolerance [3] are relevant. Interaction efficiency is improved by minimizing human interaction time. Neglect tolerance is a function of neglect time, the average time a robot can be ignored while keeping performance above some acceptable level. Larger neglect

tolerance indicates greater independence in robot operations. We apply these metrics a bit differently for robotic recon, however. Central to these differences is the distinction between *planned* and *unplanned* human interaction.

For lunar reconnaissance operations, minimizing all human interaction time may *not* translate to more efficient robot operations. In fact, it may often be more time and resource efficient to tele-operate the robot in difficult terrain than to operate autonomously. Thus, our objective is to minimize the time spent on *unplanned interventions* (such as anomaly handling).

To do this, we make use of *Mean Time to Intervene* (MTTI) [4], which is the average time spent handling anomalies that interrupt planned robot tasks. Scheduled tele-operations are not included in MTTI. We also compute *Mean Time Between Interventions* (MTBI) as the average time between unplanned interventions. Similar to neglect time, larger MTBI indicates improved human-robot performance for reconnaissance.

We measure the robot's productivity as a function of the time the robot spends on reconnaissance tasks (called *productive time*). We compare productive time to time spent on other tasks (called *overhead time*), such as waiting for a reconnaissance plan or handling problems. One metric we use is *Work Efficiency Index* (WEI) [5], which is the ratio of productive time to overhead time. We also compute the *Percentage of Time on Task* as the productive time normalized to the total elapsed time in the shift.

Performance measures used for the MER robots include the total traverse distance over the lifetime of the robot [6]. Maximizing traverse distance for such exploration is desirable as an indicator of increased productive lifespan for the robot. But longer distances traveled or drive times do not necessarily correspond to better performance during robotic recon. Specifically, some plans may have short traverses interleaved with many data collection tasks. In such cases the robot may be performing optimally even with shorter drive times and less distance traveled.

During the November 2008 ORT, for example, we observed the robot actually traveled greater distances when it was performing less well. The longest distance traveled over a shift corresponds to more human time spent unexpectedly tele-operating the robot out of difficult terrain. Comparing total distance traveled to the distance traveled performing planned tasks reveals this important distinction. Thus for robotic recon, a better indicator of good performance is the minimum distance traveled outside the plan instead of the maximum distance traveled.

Performance measures are more effective for time and resource management if good estimates of expected task times and plan durations are available. These estimates define expected baseline performance useful in interpreting measures computed in real-time. Such information can be used to assess how efficiently the robot accomplished tasks under nominal circumstances and to determine what it is feasible to get done in the time remaining under contingency conditions. Algorithms for estimating task times can benefit from work on metrics for diagnostic and prognostic technologies [7, 8]. Such algorithms address detecting problems that can affect the quality of observations, which could be useful in pruning abnormal observations from computation of typical task times.

### 3. ROBOT PERFORMANCE MEASURES

#### 3.1 Software Approach

We measure robot performance during reconnaissance operations by monitoring robot data in real-time and computing robot performance metrics from that data. Robot performance is computed remotely for use by flight operations personnel. The same robot data stream used for flight operations are used to compute performance of the robot in real-time. This includes detecting event signatures in data that affect robot performance (e.g., robot in motion). Metric values are displayed on web-based dashboards and plots for use by flight operations. Figure 3 illustrates the key components of the performance monitoring software.

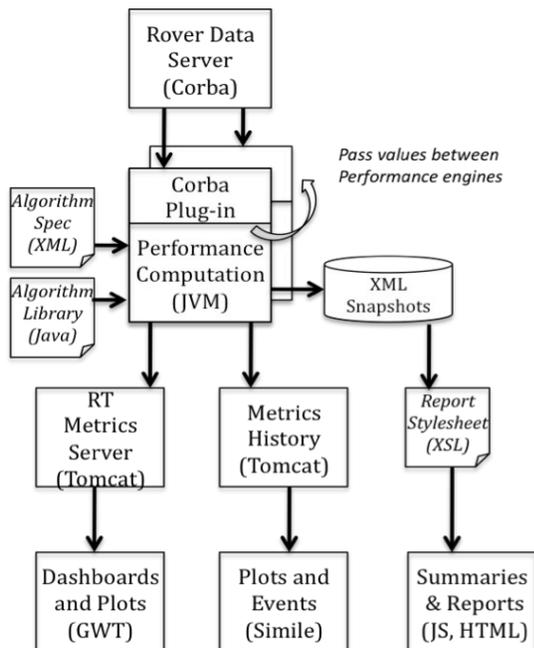


Figure 3. Components of Performance Monitoring Software

Metric algorithms are encoded as Java objects. At run-time algorithms are selected for execution and associated with robot data using configuration files. An instance of the algorithm's class is created for each connection to a robot data item, permitting algorithms to be reused across multiple metrics. Complex algorithms are composed by connecting sequences of simpler algorithms (i.e., the output of one algorithm provides input to another algorithm). Computed metrics are distributed via a real-time data server and displayed as dashboards in a web page. Performance history is provided in two ways: plots of metric values computed over time and debrief reports of summary metrics computed over a shift.

We evaluated our performance monitoring software during the June 2008 HRS field test at Moses Lake Sand Dunes, WA, to assess the feasibility of computing robot performance metrics in real time [9]. The objectives of this evaluation were to identify meaningful robot metrics, to assess whether these metrics can be computed using existing robot data, and to determine the impacts

of remoteness on the computation of robot metrics on Earth. Subsequently we supported two ORTs at NASA ARC - one in November 2008 and another in early June 2009. During these ORTs we observed how performance metrics are used in operations. We also assessed how the human team design and protocols impact robot performance, such as robot utilization and robot wait time.

Based on the results from these prior tests, we deployed and tested a revised system during the robotic recon portion of the 2009 D-RATS field test at Black Point Lava Flow, AZ. Specifically, we computed performance metrics in two time regimes – for use by flight controllers during remote real-time flight operations and for use during debrief after an operations shift. To support flight operations, we also provided web-based dashboard displays of performance metrics computed from robot telemetry data. These displays were updated automatically with the latest computed value for performance measures. To support debrief meetings, we took snapshots of metric values at the end of each shift and performed additional computations on these values to produce a debrief report spanning each shift. The debrief report was generated in a web page using the eXtensible Markup Language (XML) stylesheets and Javascript.

#### 3.2 Metrics for Real-time Flight Operations

To investigate the use of performance metrics during real-time operations, we supported two robot ground control positions [1]: the *Flight Director*, who is responsible for managing and coordinating the flight control and the *Robot Operations Coordinator*, who is responsible for the health and status of the K10 robot. In both cases, metrics are used in real-time for time and resource management.

The Flight Director primarily uses performance metrics to manage the use of time. During operations, the Flight Director is concerned with whether the robot is operating normally. This includes both whether the assigned tasks complete successfully and whether these tasks can be accomplished in the allocated time. The questions asked by the Flight Director include: How much time before an ongoing task is completed? How much time it will take to complete the unfinished tasks in the current plan? and How much of the time allocated to a plan has been expended so far? This becomes particularly important when an anomaly occurs and the Flight Director must make choices about whether to abandon some tasks.

To support the Flight Director in assessing and managing the timing of task performance, we provided a set of real-time metrics that measure how long tasks take and that compare these measures to expected performance. Specifically we computed and displayed the following five timers:

**1) Plan Timer:** The Plan Timer counts down from the expected time it will take to complete a plan while the plan is being executed. The timer is updated each time an update about plan execution status is received until the plan is done. A plan is considered done when all tasks are either successful or aborted. This timer resets when a new plan is uplinked. The expected time is computed by summing the expected time for each planned task. Expected times for sampling tasks are based on typical performance during the ORT and field test. Expected times for traverse tasks are computed for a linear path between waypoints at

default robot speed. The algorithm for the Plan Timer is shown below.

$$PlanTimer(t) = E(dt_{plan}) - (t_{current} - t_{start})$$

where

$E(dt_{plan})$  = expected time to complete plan

$t_{start}$  = time when plan begins to execute

$t_{current}$  = time of last update in plan status

**2) Plan Wait Timer:** The Plan Wait Timer increments when a plan is NOT active (i.e., no task is active, paused or pending). This includes time spent handling robot anomalies as well as idle time waiting for a new plan. This timer resets when a plan completes. Thus it measures all the “wait” time between plans. The algorithm for the Plan Timer is shown below.

$$PlanWaitTimer(t) = PlanWaitTimer(t) + dt_{elapsed}$$

where

$x$  = elapsed time between adjacent timetags

when plan not active

**3) Lidar Panorama Timer:** The Lidar Panorama Timer counts down from the expected time it will take to complete a Lidar panorama while the lidar is active. The timer is updated each time an update about the Lidar subsystem status is received until the panorama is complete. This timer resets when a new panorama begins. The expected time for the Lidar panorama task is based on typical performance during the ORT and field test. The algorithm for the Lidar Panorama Timer is shown below.

$$LidarTimer(t) = E(dt_{panorama}) - (t_{current} - t_{start})$$

where

$E(dt_{panorama})$  = expected time to complete panorama

$t_{start}$  = time when Lidar panorama subsystem goes active

$t_{current}$  = time of last update of Lidar subsystem status

**4) PanCam Timer:** The Panoramic Camera (PanCam) Timer counts up while a PanCam panorama is being taken. For the 2009 field test at Black Point Lava Flow, there are five different types of panorama: (1) Medium Width Image, Low Resolution, (2) Medium Width Image, Medium Resolution, (3) Wide Image, Low Resolution, (4) Wide Image, Medium Resolution, and (5) Narrow Image, High Resolution. Each of these types of panoramas take a different amount of time. Since the field test at Black Point Lava Flow was the first field test where we have used variable width and resolution PanCam imaging, we employ a “count up” timer to collect data on typical task times at each resolution. The algorithm for the PanCam Timer is shown below.

$$PanCamTimer(t) = PanCamTimer(t) + dt_{elapsed}$$

where

$x$  = elapsed time between adjacent timetags

when taking panorama

**5) MicroImager Timer:** The MicroImager Timer counts up while a MicroImage is being taken. Microimaging takes between 10 and 20 seconds. Because this task is so short, counting down from some expected time was not deemed useful. The algorithm for the MicroImage Timer is shown below.

$$MicroImageTimer(t) = MicroImageTimer(t) + dt_{elapsed}$$

where

$x$  = elapsed time between adjacent timetags

when microimaging

The Robot Operations Coordinator uses “low-level” performance metrics to manage robot resources such as remaining battery power. Such measures are needed to anticipate when robot maintenance is needed. Some batteries on the K10 robot provide feedback about remaining capacity and some do not. For batteries that are instrumented, we provide displays of the average battery capacity for each battery controller. Each controller manages eight batteries. For batteries that are not instrumented (such as the Lidar battery) we track battery usage (i.e., total runtime of the instrument using the battery) to give the Robot Operations Coordinator an idea of how much capacity remains. Utilization is tracked throughout the shift over multiple sampling intervals. When a Lidar battery is swapped out, we reset Lidar Run Time. The algorithm used for Lidar Run Time is shown below.

$$LidarRunTime(t) = t_{end} - t_{start}$$

where

$t_{start}$  = time when Lidar panorama subsystem goes active

$t_{end}$  = time when Lidar panorama subsystem goes inactive

Finally we compute data communication quality metrics for use by all flight controllers. During the field test at Moses Lake, we observed two days with significantly degraded communication. The metric for communication quality during this field test was a count of the number of times the ground lost communication with the remote robot (called a *data dropout*, or *Loss of Signal[LOS]*) during a support period (for this test we compute a count over the shift). Larger dropout counts indicated more data were unavailable for computation, potentially impacting the accuracy of metric values. The duration of dropouts varied significantly at Moses Lake, however, prompting the definition of a second metric to compute the percentage of a support period (i.e., shift) that was spent without communication. This permits estimating the amount of data not available to flight operations and not included in metrics computed during real-time operations. The algorithms for both these metrics are shown below.

$$DataDropout(tp) = \begin{cases} t & \text{if } tp_i - tp_{i-1} > 10 \text{ sec} \\ f & \text{if } tp_i - tp_{i-1} \leq 10 \text{ sec} \end{cases}$$

where

$tp_i$  = time of pose message  $i$

$$\%ShiftInLOS(t_i) = [TimeInLOS(t_i) / TimeInShift(t_i)] * 100.0$$

where

$TimeInLOS(t_i)$  = Elapsed time in LOS at  $t_i$

$TimeInShift(t_i)$  = Elapsed time in shift at  $t_i$

### 3.3 Metrics for Shift Debrief

Operationally, the ground control team holds a debrief meeting immediately after each shift (contiguous period of operations). We compute performance metrics over the course of each shift for building a debrief report. The debrief report has three sections: (1)

a robot performance summary, (2) an anomaly summary, and (3) an event log. The performance summary provides metrics about robot productivity, task breakdown, and data collected during the shift. The anomaly summary provides metrics about unplanned interventions in robot operations, problems experienced by the robot, and loss of communication. The event log interleaves log notes by the flight team with events detected automatically in the robot telemetry stream.

Three metrics are computed to assess robot productivity – time spent performing planned tasks (called productive time), time spent doing activities other than planned tasks (called overhead time), and the ratio of these two measures (called Work Efficiency Index; [5]). The algorithm used for each of these metrics is shown below.

$$WEI(t_i) = PT(t_i)/OT(t_i)$$

where

*PT(t<sub>i</sub>) = time executing planned tasks*

*OT(t<sub>i</sub>) = time outside planned tasks*

To assess task performance we compute the breakdown across the shift of robot drive time, time taking Lidar, time taking each of the five types of PanCam, and time microimaging. We also measure the number of samples taken by each type of instrument over the shift. Finally, we compute the total distance traveled by the robot during the shift. The algorithms used for these robot performance metrics are shown below.

$$RunTime(t) = \sum_{i=1} t_i - t_{i-1} \quad \text{if } t_i - t_{i-1} < 10 \text{ sec}$$

where

*t<sub>i</sub> = time of pose message*

$$DriveTime(t) = \sum_{i=1} t_i - t_{i-1}$$

where

*t<sub>i-1</sub> = last time locomotor or navigator went active*

*t<sub>i</sub> = time both locomotor and navigator go inactive*

$$DistanceTraveled(x,y) = \sum_{i=1} \sqrt{[(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2]}$$

if  $d(x,y) > 0.025m$

where

*(x<sub>i</sub>,y<sub>i</sub>) = current rover pose estimate*

*(x<sub>i-1</sub>,y<sub>i-1</sub>) = previous rover pose estimate*

To assess anomalies in robot operations, we detect unplanned interventions in robot operations and use them to compute the Mean Time To Intervene (MTTI) and the Mean Time Between Interventions (MTBI). We show how many of each type of problem intervention occurred during the test, and the percent of the shift spent on each type of intervention. We also summarize communication quality by measuring the total number of LOS periods and the percentage of the shift spent in LOS. The algorithms for MTTI and MTBI metrics are shown below. The algorithms for communication quality were described previously.

$$MTTI(t_i) = TI(t_i)/CI(t_i)$$

where

*TI(t<sub>i</sub>) = time spent intervening upto t<sub>i</sub>*

*CI(t<sub>i</sub>) = number of interventions by t<sub>i</sub>*

$$MTBI(t_i) = TBI(t_i)/CBI(t_i)$$

where

*TBI(t<sub>i</sub>) = total time between interventions upto t<sub>i</sub>*

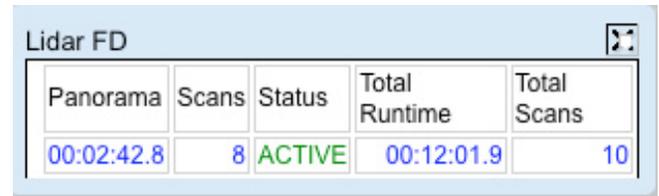
*CBI(t<sub>i</sub>) = number of periods between interventions by t<sub>i</sub>*

The event log combines information derived from the robot telemetry stream with log entries made by users. Events detected automatically include (1) Acquisition of Signal (AOS) and Loss of Signal (LOS), (2) instrument sample start and end, (3) problem start and end, and (4) plan uplink, start, and end. Problems reported include (1) emergency stop, (2) critical failure in robot subsystem (locomotor, navigation, or plan executor), (3) joint failure, and (4) navigation position error (usually due to a stuck wheel). These are problems where intervention by a person is likely required to fix the problem. User log entries are made from our real-time display. They are timestamped at the time of log entry. User log entries are distinguished from detected log entries using italics text in the display.

#### 4. RESULTS

The performance monitoring software was used by K10 ground control during the robotic recon portion of the 2009 D-RATS field test at Black Point Lava Flow, AZ. In this section we summarize the results from using these metrics during field test operations.

Instrument timers (Figure 4) were used in a number of ways during real-time operations. First, they were used to make better estimates of task duration for data acquisition tasks. Accurate estimates of the time to acquire the five types of PanCam panoramic images and the three types of Lidar scans were not available prior to the field test. We averaged sample collection times during operations early in the test to determine more accurate estimates. These estimates were used by the science team when building plans. They also were used by the Flight Director when monitoring robot progress.



Panorama	Scans	Status	Total Runtime	Total Scans
00:02:42.8	8	ACTIVE	00:12:01.9	10

Figure 4. Example of Lidar Task Timer Display

Second, the elapsed data acquisition time from an instrument timer was compared to the expected acquisition time for the sample type to determine the time remaining in an acquisition task. The medium and high resolution PanCam samples took between 10 and 17 minutes to perform and the Flight Director frequently used these timers to determine time to task completion,

Third, the instrument timers were used to detect data acquisition problems. These problems were evidenced by a larger than expected elapsed time with no indication that a sample had been taken (i.e., the sample count did not increase). The Lidar instrument had difficulty in completing a panorama on multiple occasions due to the extreme thermal conditions and the Lidar timer was useful in detecting these problems quickly.

Finally, the Robot Operations Coordinator used the total Lidar runtime to estimate when to swap out the Lidar battery, since direct sensing of battery level was not available. During the field test we added the ability for the user to reset this timer after a battery swap.

Plan timers were used by the Flight Director to identify the currently active task and when it was marked complete or aborted, and to determine what type of data was being acquired. The plan timers were helpful in determining when a plan had gone beyond the allocated time, but did not provide the Flight Director with sufficient timing information to detect when plan execution was getting behind early enough to take action before the allocated time was expended. When plan execution did get behind, the plan timer also did not aid the Flight Director in determining whether some portion of the remaining tasks might be completed in the time left. We developed new metrics during the field test to address these needs. Specifically we computed the time needed to complete the ongoing task and all pending tasks using estimates of task time previously measured. Using this information, the Flight Director could assess if adequate time remained and, if not, could inspect estimated task times to aid in selecting which remaining tasks to perform (a form of contingency re-planning). For the field test, this information was provided as a snapshot display from a checkpoint of metrics and thus did not update as plan execution continued. For future tests, we believe a version of this display that updates with progress on the plan would be preferable. Figure 5 shows an example of the display used during the field test for these additional plan timing metrics.

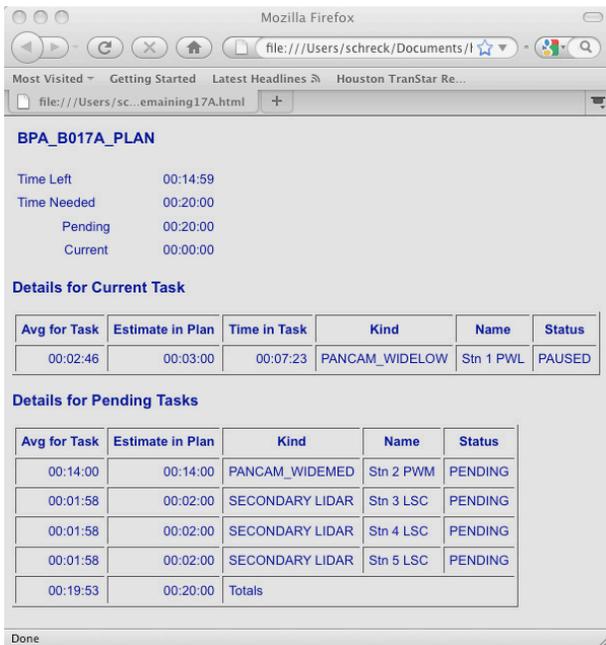


Figure 5. Display of Time Remaining in Plan 17A on June 20

High environmental temperatures increased the risk of robot subsystems overheating during operations. The metrics for battery management were used by the Robot Operations Coordinator primarily to monitor battery temperature for potential overheating. To support this task, we computed the maximum temperature observed for each group of eight batteries and compared it to the current temperature. The Flight Director felt the thermal information currently provided should be supplemented with summary metrics of thermal performance,

Communication quality varied significantly over the course of the field test. The average percentage of time spent in LOS for the entire field test was 13.4%, and varied from 1% to 38% of the day in LOS (Figure 6). Our metric for communication quality defined LOS as a dropout of all robot data. It was not uncommon, however, to experience dropped data messages without losing all robot data. As a result, the LOS metric is a conservative estimate of the quality of communication that can underestimate the impact of communication anomalies on data availability and quality of metrics. The Flight Director supplemented the LOS metrics with the message dropout warnings available for the individual robot subsystems to detect degraded communication due to dropped messages.

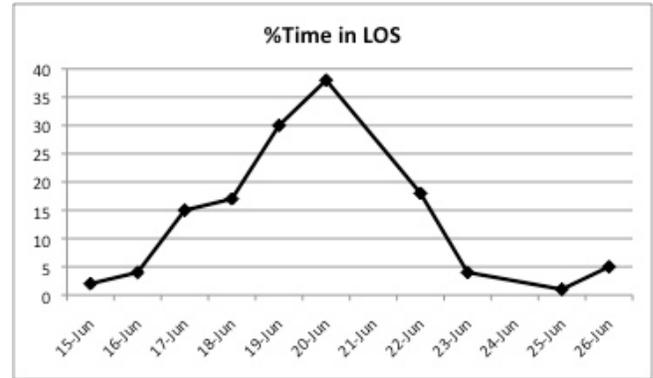


Figure 6. Daily Percentage of Time in LOS for Field Test

We produced a summary report at the end of each day using the metrics computed in real-time. These reports are intended to clarify how the robot spent its time, what data were collected, and what problems were encountered. These reports also should aid in identifying trends in expected robot performance over the field test. The summary measures in the debrief report identify how well the robot performed its tasks (see Figure 7 for an example of robot productivity on June 25) and what problems occur that affect that performance (see Figure 8 for an example of robot anomalies on June 25). An event log provides the details about what circumstances contributed to these summary measures. For example, one metric in the anomaly summary is the mean time spent by personnel intervening unexpectedly in robot operations. The log can be inspected to see exactly when samples were taken or when interventions were made. By inspecting adjacent events, the user may gain additionally insight into the operation.

These reports were used as supplements to the console notes taken by the flight controllers. Both the Flight Director and Robot Operations Coordinator felt these reports would be more useful if flight operations personnel could edit them. The types of editing mentioned includes removing some data points from metric computations and adding comments after operations when reviewing and analyzing performance. At the end of the test these

reports also represent a mission performance summary of robot productivity and robot reliability.

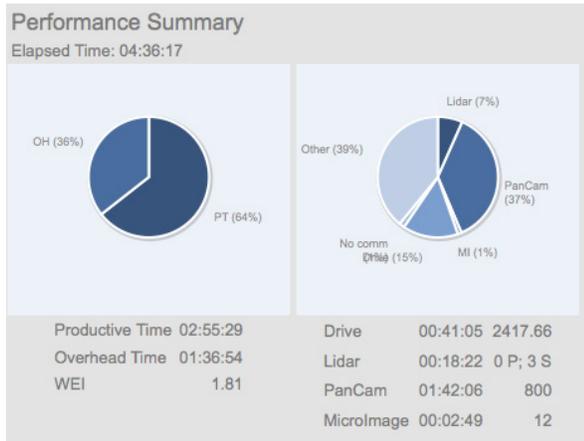


Figure 7. K10 Robot Productivity on June 25

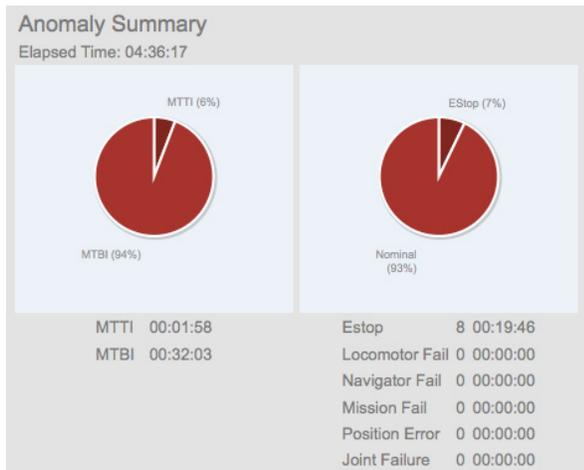


Figure 8. K10 Robot Anomalies on June 25

The average K10 robot productivity for the entire field test at Black Point was 37% productive time, 63% overhead time, and an average WEI of 0.73. Figure 9 shows these productivity metrics computed in real-time for each day of the field test.

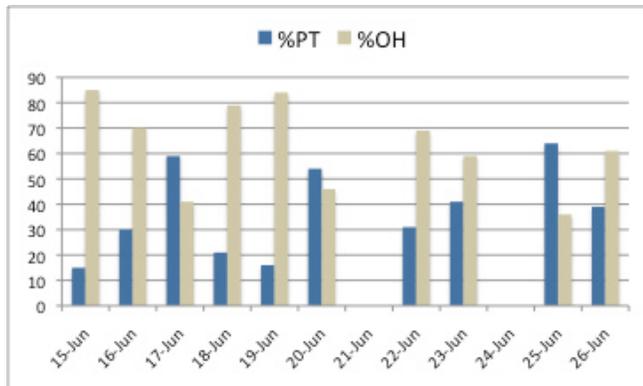


Figure 9. Daily K10 Robot Productivity Metrics for Field Test

We further decomposed robot time into the percentage of time spent on each type of task. Figure 10 shows this breakout from

June 17-26. We eliminated data on June 15-16 because of an error in the drive time computation used in real-time for those days.

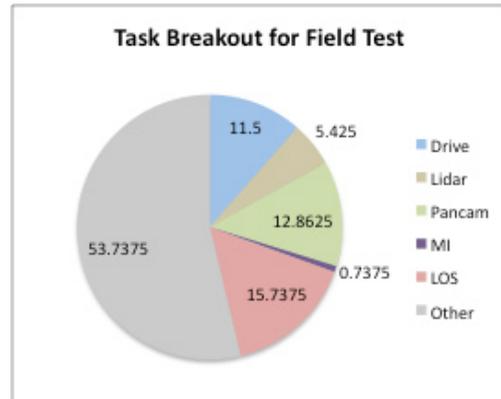


Figure 10. K10 Robot Task Breakout for the Field Test

The average MTTI computed in real-time for the entire field test at Black Point was 5.6 minutes, ranging from a minimum of 1.6 minutes to a maximum of 17.9 minutes. The average MTBI was 24 minutes, ranging from a minimum of 5.5 minutes to a maximum of an hour. Figure 11 shows these reliability metrics computed in real-time for each day of the field test.

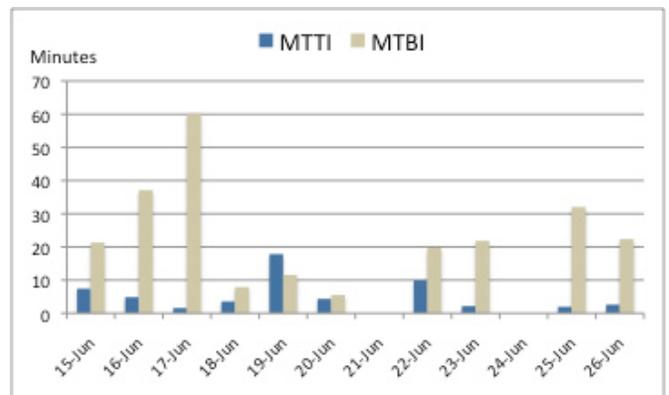


Figure 11. Daily K10 Robot Reliability for the Field Test

## 5. CONCLUSIONS

We have described an approach for in-line computation of robot performance metrics to aid human-robot interaction during remote operation of robots in space. Results of evaluating our approach during recent field tests with the K10 robot indicate real-time computation of robot performance can aid both robot operations and debrief after operations.

Plan and task timing metrics were used frequently during operations. These metrics were most meaningful when used in the context of expectations. Instrument timers were combined with knowledge of expected data acquisition times to monitor progress on the task and to detect when these task were not collecting data as expected. Similarly plan timers were combined with estimates of plan duration to monitor progress on the plan and to detect when the robot was getting behind. New metrics were identified and computed during the field test that compare the time left to complete the plan with estimates of the time needed to complete

the remaining tasks. This information was used to determine whether adequate time remained to complete a plan and, if not, which of the remaining tasks should be performed. In all these cases, the expected task timing also was computed by the performance software. Comparing actual timing to expected timing is useful when inspecting performance after the mission as well. During the field test we computed the ratio of actual time to complete a plan with the estimated time to complete a plan. Using this metric we determined that 11 of 20 plans taken to completion were performed within the allocated time.

Operational use of WEI and Percentage of Time on Task to measure robot productivity indicates that the Percentage of Time on Task is more meaningful in real-time. WEI is difficult to interpret for real-time use. When overhead time is very small, WEI can be very large (or can be undefined if overhead is zero). The meaning of such large numbers is not clear. By normalizing productive time to total time in operations (i.e., shift time), the Percentage of Time on Task is guaranteed to vary between 0 and 100, ensuring greater consistency across operations and shifts.

Communication quality varied significantly over the course of the field test, ranging from 1- 38% of daily operations spent out of communication. Additionally we observed frequent dropped messages that affected data availability and the resulting quality of metrics. As a result of this loss of data, we observed small inaccuracies in statistics on task durations and data sample counts due to communication anomalies. Additional metrics are needed that characterize how dropped messages affect data availability. The prevalence of dropped messages also indicates that metrics computed using this lossy data are subject to error due to missed data and could benefit from algorithms that consider the quality of the data messages used to compute them.

## 6. FUTURE WORK

Early detection of plan threats give ground control more flexibility in re-planning because more time and resource remain than if the threat is detected late. Plan threats include getting behind when performing a plan, using more resource (such as battery power) than planned, or losing robot capability that affects plan completion. We plan to investigate metrics for early detection of plan threats. For example, can we detect a robot getting behind as a divergence between the time remaining in the plan and the time needed to complete the plan?

We also plan to investigate approaches for detecting when dropped messages impact quality of metrics. This includes performance algorithms that consider the quality of the data messages used to compute them.

We believe the performance data we have computed over multiple NASA field tests can be useful for future field tests. It can be used to characterize typical robot performance for terrain types or specialized operations and thereby improve our interpretation of future robot performance (i.e., was this typical robot performance for this type of terrain?). It also can be used to establish realistic expectations when designing the activities for future field tests.

## 7. ACKNOWLEDGMENTS

This research was funded by the NASA Small Business Innovative Research program, Human-Robotic Systems project and ESMD Analogs program. We would like to thank Rob Landis, Timothy Kennedy, and Frank Jurgens for their feedback about flight control metrics. Thanks also to David Lees and Mark Allen for their assistance in integrating the performance monitoring software with the K10 ground data system, to Estrellina Pacis for her assistance in defining metrics for operations assessment, and to Matt Deans for information about the robotic recon planning process.

## 8. REFERENCES

- [1] Fong, T., M. Bualat, M. Deans, M. Allan, X. Bouyssounouse, M. Broxton, L. Edwards, R. Elphic, L. Fluckiger, J. Frank, L. Keely, L. Kobayashi, P. Lee, S. Y. Lee, D. Lees, E. Pacis, E. Park, L. Pedersen, D. Schreckenghost, T. Smith, V. To, and H. Utz. *Field Testing of Utility Robots for Lunar Surface Operations. AIAA Space 2008*. San Diego, CA.
- [2] Jacob Crandall, Michael Goodrich, Dan Olsen, Jr., and Curtis W. Nielsen, "Validating Human-Robot Interaction Schemes in Multitasking Environments", *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, Vol. 35, No. 4, July 2005
- [3] Jacob Crandall and M. L. Cummings. Developing Performance Metrics for the Supervisory Control of Multiple Robots. *Human-Robot Interaction 2007*. March 2007.
- [4] J. Arnold, "Towards a Framework for Architecting Heterogeneous Teams of Humans and Robots for Space Exploration", M.S. Thesis, Dept. of Aeronautics and Astronautics, Massachusetts Institute of Technology. 2006.
- [5] Mike Gernhardt, Work Efficiency Indices. Presentation at Johnson Space Center. November 15, 2005
- [6] Edward Tunstel, "Performance Metrics for Operational Mars Exploration Rover," *Journal of Field Robotics*, Volume 24 Issue 8-9, 651 – 670, September 2007.
- [7] Tolga Kurtoglu, Ole J. Mengshoel, and Scott Poll, A Framework for Systematic Benchmarking of Monitoring and Diagnostic Systems," *2008 International Conference on Prognostics and Health Management Proceedings*, October 6-9, 2008.
- [8] Abhinav Saxena, Jose Celaya, Edward Balaban, Kai Goebel, Bhaskar Saha, Sankalita Saha, and Mark Schwabacher, "Metrics for Evaluating Performance of Prognostic Techniques," *2008 International Conference on Prognostics and Health Management Proceedings*, October 6-9, 2008.
- [9] Debra Schreckenghost, Terrence Fong, Tod Milam, Estrellina Pacis, and Hans Utz. *Real-time Assessment of Robot Performance During Remote Exploration Operations*. IEEE Aerospace Conference. Big Sky, MT. March 2009.