TELE-OPERATED LUNAR ROVER NAVIGATION USING LIDAR

Liam Pedersen[1], Mark Allan[2], Hans Utz[3], Matthew Deans[4], Xavier Bouyssounouse[5], Yoonhyuk Choi[6], Lorenzo Flückiger[7], Susan Y. Lee[8], Vinh To[8], Jonathan Loh[9], William Bluethmann[10], Robert R. Burridge[10], Jodi Graf[10], and Kimberly Hambüchen[10]

[1]Carnegie Mellon University / NASA Ames Research Center, Moffett Field, CA, USA, Liam.Pedersen@nasa.gov
[2]SGT / NASA Ames Research Center, Moffett Field, CA, USA, Mark.Allan@nasa.gov
[3]Carnegie Mellon University / NASA Ames Research Center, Moffett Field, CA, USA, Hans.Utz@nasa.gov
[4]NASA Ames Research Center, Moffett Field, CA, USA, Matthew.Deans@nasa.gov
[5]NASA Ames Research Center, Moffett Field, CA, USA
[6]KAIST, Seoul, South Korea
[7]Carnegie Mellon University / NASA Ames Research Center, Moffett Field, CA
[8]SGT / NASA Ames Research Center, Moffett Field, CA, USA
[9]University of California Santa Cruz, Santa Cruz, California, USA
[10]NASA Johnson Space Center, Houston, TX, USA

ABSTRACT

Near real-time tele-operated driving on the lunar surface remains constrained by bandwidth and signal latency despite the Moon's relative proximity.

As part of our work within NASA's Human-Robotic Systems Project (HRS), we have developed a stand-alone modular LIDAR based safeguarded tele-operation system of hardware, middleware, navigation software and user interface. The system has been installed and tested on two distinct NASA rovers–JSC's Centaur2 lunar rover prototype and ARC's KRex research rover–and tested over several kilometers of tele-operated driving at average sustained speeds of 0.15 - 0.25 m/s around rocks, slopes and simulated lunar craters using a deliberately constrained telemetry link.

The navigation system builds onboard terrain and hazard maps, returning highest priority sections to the off-board operator as permitted by bandwidth availability. It also analyzes hazard maps onboard and can stop the vehicle prior to contacting hazards. It is robust to severe pose errors and uses a novel scan alignment algorithm to compensate for attitude and elevation errors.

Key words: navigation; lunar tele-operation, Centaur2, hazard detection, RAPID, NASA telemetry standard, point cloud alignment, LIDAR, roughness estimation.

Figure 1: The Centaur2 rover at NASA JSC's Rockyard test site. Note the modular navigation mast installed on the right

Figure 2: NASA ARC's KRex research rover

1. INTRODUCTION

Near real-time tele-operated driving on the lunar surface remains constrained by bandwidth and signal latency. Proposed robotic lunar science missions [1] require vehicles to drive 3-5 km in 4-7 days to accomplish their objectives. NASA's Human Architecture Team recommends a maximum rate of 384 kbps for downlink and 10 kbps rate for uplink [2].

This work documents our effort to demonstrate a modular system for tele-operating a lunar rover over a kilometer subject to a 1 Mbps bandwidth constraint for *uncompressed* data. The system consists of sensors, computation, middleware, navigation software and a UI. It is deployed on two distinct NASA robots - JSC's Centaur2 Lunar rover prototype (Figure 1) and ARC's KRex research platform (Figure 2).

This work builds on our earlier LIDAR navigation stack [3], first developed to navigate CMU's SCARAB rover [4] in complete darkness, and later demonstrated on NASA's Lunar Electric Rover (now called the Space Exploration Vehicle or SEV) during the 2011 DRATS simulated lunar operations test [5]. The algorithms used have direct heritage from Simmons *et al* classic RANGER architecture [6], derivatives of which have since been used by stereo-vision based planetary rovers, and the work of Thrun *et al* for the DARPA Grand Challenge [7, 8].

The subsequent sections detail the navigation sensor and computation stack and its integration to the Centaur2 platform, the RAPID middleware and telemetry standard developed in conjunction with this project, the mapping algorithms, and user interfaces. Preliminary results from a June 2012 demonstration of kilometer scale driving under the constrained bandwidth are also presented.

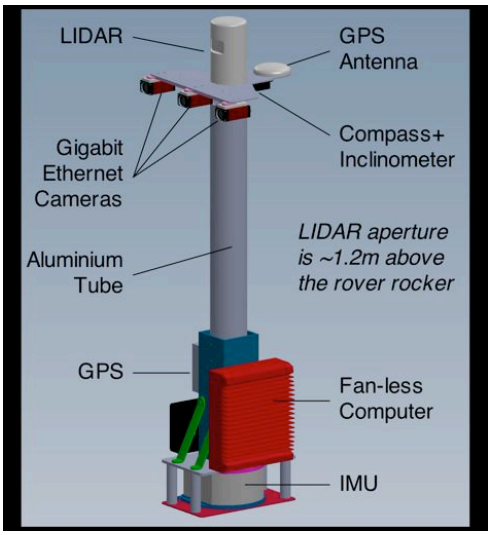

Figure 3: Navigation sensor mast, installed on Centaur2 and KRex

2. HARDWARE

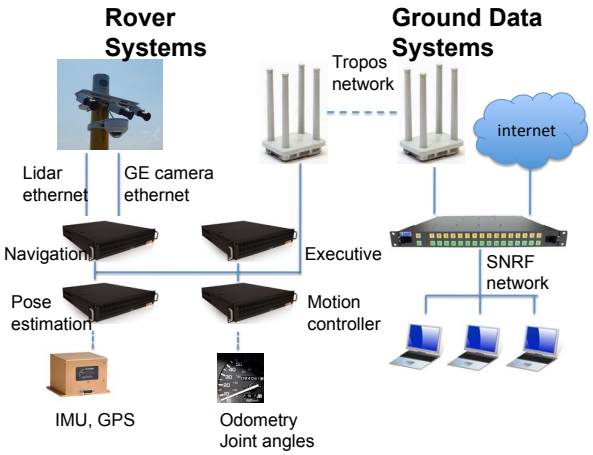

Figure 4: Centaur2 hardware and networking

The Centaur2 navigation "sensor mast" (Figure 3) is a standalone system that can be easily ported between rovers of similar size. It includes terrain and pose sensors plus computation in a single package.

The Velodyne HDL-32E LIDAR is the primary terrain mapping and navigation hazard detection sensor. Thirty-two beams provide simultaneous range measurements from -30° to +10° in elevation above and below the sensor, with a complete 360° scan around the rover 10 times per second. We subsample the data in time.

Forward-looking stereo cameras (Table 1) provide denser range measurements ahead, and in particular include the Velodyne blind spot immediately in front of the rover caused by the minimum elevation angle of the lidar. This additional coverage is useful for looking down over cliffs or on steep slopes.

The fan-less navigation computer (Table 2) has separate network connections to the LIDAR and Gigabit Ethernet cameras to isolate their traffic from the main rover LAN (Figure 4). On Centaur2, additional computers provide pose estimation (Table 3), executive and motion control services. Frequent (every 5 minutes) NTP synchronization is required between the navigation and pose estimation computers. A Tropos wireless network links the rover LAN to ground data systems on NASA's network via a specialized router configured to enforce a 1 Mbps connection limit and (optionally) introduce a user specified communications latency.

Table 1: Stereo cameras

Stereo baseline	30cm
Lens field of view	82° x 67°
Lens focal length	3.5mm (C-mount)
Camera sensor	color 1388 x 1038 pixels
Camera interface	Gigabit Ethernet

Table 2: Navigation computer

Processor	Intel Core i7-620M @ 2.66GHz
Storage	120GB SSD
Memory	4GB DDR3
Cooling	passive fan-less
Serial ports	3 x RS-232 + 1 x RS-232/422/485
Network ports	5 x Gigabit Ethernet
Physical	240x76x195mm - 2.7 kg

Table 3: Pose estimation systems and *estimated* accuracies

	KRex	Centaur2
Position	Novatel OEM 4 DGPS	Topcon GPS
	5cm	3.5m horizontal 5m vertical
Orientation	HG1700 FOG	Crossbow VG700
	0.1° roll,pitch 0.3° yaw	2.5° roll,pitch $> 5°$ yaw Estimated from GPS

3. ARCHITECTURE AND MIDDLEWARE

The safe-guarded tele-operations system is designed at its core as a distributed system of decoupled modules running on multiple machines, both onboard and off-board. Communications between modules and between robot and ground-control uses NASA's inter-center Robot API Delegate (RAPID) interfaces and message specifications [9], built on the Data Distribution Service (DDS) open standard for data-centric publish-subscribe architectures [10], and developed in partnership with this project.

RAPID/DDS accommodates the large differences in bandwidth and latency of the on-robot versus robot-ground communication links, identified by unique DDS domain ID's. The Quality of Service (QoS) parameters for each domain are uniquely adjusted to the characteristics of each link. A key parameter is reliability. Certain data products (e.g. pose) are quickly stale, and bandwidth should not be wasted on protocols to ensure delivery of all pose data offboard. Conversely, maps data *must* be reliably sent from the mapping module to the path evaluation module in a timely manner.

To date, we have not taken full advantage of the DDS middleware capabilities; for simplicity some data streams (maps) are not yet split into on-board and off-board streams. Additional DDS features for future investigation include lossless data compression, header compression, and rate limiting.

The mapping and path-analysis modules are built within our Service Oriented Robotics Architecture (SORA) framework [11], consisting of a number of distinct modules that subscribe to common data products (sensor measurements, pose estimates) via the RAPID/DDS communications layer, and respond to remote procedure calls from other modules or the User Interface (UI).

The inherently distributed nature of SORA and the RAPID message specifications reduced the complexity of integrating the navigation hardware and software to Centaur2, which has its own distinct telemetry, pose estimation, sequencing, and motion control subsystems running on different operating systems (the navigation stack runs on Linux, the pose estimation and sequencing on Windows, and motion controllers on VxWorks). Through implementation of a bridge to the commonly agreed RAPID API the two architectures could be tied together.

The power of the RAPID and SORA based approach is nicely illustrated by the fact that the system was developed as an inter-center project, with the navigation stack being deployed and tested on two different NASA robot platforms simultaneously (Centaur2 [12] and KRex [13]), due to availability of the robotic platforms at the different NASA centers. Centaur2 and KRex feature very different control architectures and have different hardware characteristics.

4. HAZARD DETECTION AND MAPPING

The hazard detection and mapping subsystem, shown in Figure 5, combines range measurements $r_i(t)$ with the corresponding rover pose estimate $\vec{P}(t)$ (interpolated from the previous and subsequent pose estimation system outputs) to compute the point cloud $\vec{x}_i(t)$. A *small* pose correction $d\vec{P}(t)$ is computed to align the point cloud with the current topography map. The aligned points $\hat{\vec{x}}_i(t)$ update their corresponding cells in the statistics accumulator map Σ while $d\vec{P}(t)$ is retained for future updates.

The statistics accumulator map Σ, represented as a scrolling rectangular array for constant time determination of point-cell correspondences, tracks the correlation moments, min and max heights and other (minimal) information needed to robustly compute the topography and height difference maps. Accumulated statistics from the past are progressively de-weighted in favor of current data to account for changes in the environment and accumulated pose errors. For details see [3].

Height difference and subsampled topography maps are computed twice per second from the accumulator map, and merged with a user-specified map to create the combined navigation map, consisting of *traversability*, *certainty* and *roughness* values within [0,1] (equation 1). Traversability can be interpreted as an inverse cell traversal cost for path evaluation and planning purposes. Hazards that must be avoided have zero traversability (∞ cost). Cells with certainty less than 0.5 are considered unexplored, and also treated as hazards for the purposes

of local path evaluation. Nearby unexplored regions are due to occlusions, steep drop offs or holes in the ground.

$$trav = min(tr_{slope}, tr_{height\ diff}) \quad (1)$$

$$tr_\theta = ls(\theta, \theta_{caution}, \theta_{hazard}) \quad (2)$$

$$ls(x,a,b) = \begin{cases} 0 & x > b \\ 1 & x < a \\ (b-x)/(b-a) & o.w. \end{cases} \quad (3)$$

Certainty is similarly computed based on the accumulated weight of points assigned to each cell in the accumulator map. This works when the vehicle is in motion and the points cover the full cell area, but breaks down when stationary and the same points are repeatedly added to a cell.

The combined map is broken into smaller map tiles that are individually exported over the DDS telemetry channel and individually displayed in the UI. Only map tiles that change are exported, and they are prioritized by proximity to the rover's current position. Lower priority tiles are discarded when bandwidth is constrained. The UI displays all tiles received, including those corresponding to areas left behind the rover and no longer being actively mapped.

Path checking is efficiently (albeit conservatively) done on a 2D C-space map, wherein hazards and unexplored areas are expanded by the robot radius. Our map representation is such that we can do this with the OpenCV erode function.

Pose computed from GPS, or infrequent landmark observations, contains discontinuities that corrupt the maps, challenge the point cloud alignment, and lead to apparent changes in hazard locations with respect to the rover. To prevent this, relative position (from odometry only) is used in the mapping and collision checking modules. This approach is based on an assumption that pose inaccuracies accumulate sufficiently slowly to avoid major map distortions.

The rover maintains the transform between its relative (odometry only) and global (using GPS) pose estimates. Each exported map tile is tagged with its global pose so that the UI will display it correctly relative to prior maps and overhead images of the site. This results in the tiles shifting slightly with respect to each other as the two different pose estimates diverge, but the tiles remain self-consistent and locally consistent.

4.1. Map alignment

The map alignment algorithm, illustrated in Figure 6, uses Point-To-Plane ICP with small angle trigonometry approximations to efficiently compute the rotation and translation that minimizes the sum of squared normal

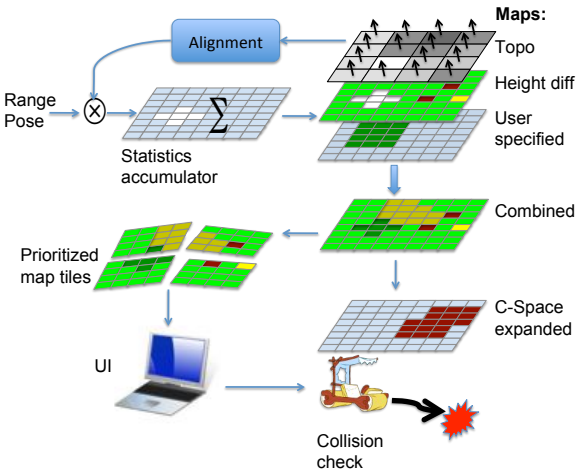

Figure 5: Mapping Algorithm. White map cells have no data (certainty < 0.5), green indicates drivable (traversability = 1), red hazards (traversability = 0), and yellow caution (traversability in (0,1)

distances between points and their corresponding planes [14]. The algorithm proceeds as follows:

for all points $\vec{x_i}$ in point cloud **do**
  get corresponding map cell *k*
  if ACCEPT($\vec{x}$,*k*) **then**
    add cell center $\vec{o_k}$, normal $\vec{n_k}$, and normal std dev σ_k to lists $\{\vec{o_k}\}$, $\{\vec{n_k}\}$, $\{\sigma_k\}$
  end if
end for
COMPUTE

$$d\vec{P} = (dz, droll, dpitch)^T \quad (4)$$

$$= argmin \sum_k (d\vec{P} * (\vec{x}_k - \vec{P}) + \vec{P} - \vec{o}_k) \cdot \vec{n}_k / \sigma_k \quad (5)$$

The correspondence acceptance test above requires that the cell correspond to planar terrain, the point be close to the cell center, and time between point acquisition and last update of cell statistics be limited. Cells are considered planar if the standard deviation (in the normal direction) of all prior points added to the cell is a fraction of the cell width.

Only one iteration is required, since point-cell correspondences barely change after application of $d\vec{P}$. As such, the algorithm is fast enough to apply to every input scan. Each time $d\vec{P}$ is accumulated and applied to the next point cloud prior to alignment.

4.2. Height Difference Maps

Rocks higher than the vehicle ground clearance (30cm for Centaur2) are hazards. A map cell is thus considered untraversable if two points within ϵ horizontal distance

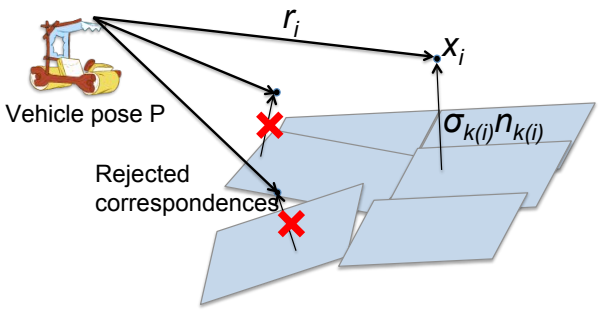

Figure 6: Point cloud alignment to terrain map

of each other differ in height by by more than a threshold δ. Naively applying this test results in an unacceptable number of false positives due to pose errors, *even after applying map registration*.

Thrun *et al* [8] developed a statistical test for hazardous height differences that is robust (up to a point) to pose noise. Essentially, given two points $(X_k^i, Z_k^i, Z_k^i)^T$ and $(X_m^j, Z_m^j, Z_m^j)^T$ acquired at times k and m, then

$$Pr(|Z_k^i - Z_l^j| > \delta) > \pi \tag{6}$$

if and only if

$$|Z_k^i - Z_l^j| - \delta > T \tag{7}$$

Where T is of form

$$T = \sqrt{a|k-l| + b} \tag{8}$$

and depends on the corresponding range measurements and the statistics of the pose noise.

To compute whether a cell passes the obstacle test (6) it is sufficient to track the minimum and maximum heights of corresponding points, along with their associated range measurements and acquistion times.

This test has the drawback of disregarding obstacles just below the detection threshold. Tracking the actual height differences would enable us to flag cautionary obstacles, and generalize the map for use with different bottom clearances (Centaur2 can adjust this). We invert (7) to track a statistically robust *lower bound* δ_i on the height differences at each cell i (or between cells ϵ apart):

$$\delta_i = max(\delta_i^{old}, max(|Z_k^i - Z_l^\epsilon| - T)) \tag{9}$$

This can be reduced (approximately) on slopes θ by $\epsilon cos(\theta)$, the height difference due solely to terrain inclination.

5. USER INTERFACE

The user interface draws components from two projects. The first, PIGI (Predictive, Interactive Graphical Interface) provides interfaces for teleoperating robots over time-delayed and limited bandwidth links. Second, VERVE (Visual Environment for Remote & Virtual Exploration), provides 3D displays to visualize robot telemetry in real time. Both projects are part of the NASA Ensemble platform, a component-based plugin architecture based on the Eclipse RCP (Rich Client Platform) and implemented in Java.

The Centaur2 driving interface is provided by PIGI, a suite of tools designed to keep the human in the loop as much as possible while taking advantage of short-term robot autonomy [15]. A typical command cycle involves the operator placing a waypoint, then sending that command to a behavioral simulator which predicts the robot's drive path. If the operator is satisfied with the path, the command is issued to the robot's task queue, and the operator begins planning the next waypoint. PIGI integrates with VERVE to display the waypoints and predicted drive paths in the 3D view, and also includes user interface elements for manipulating the robot's task queues and viewing robot status information.

The VERVE 3D components are designed for high fidelity visualization of multi-robot operating scenarios. VERVE supports large scale terrain rendering through the use of geometry [16] and texture clipmaps [17] for operations in outdoor environments. Multiple sets of tiled DEMs (Digital Elevation Model) and orthorectified satellite imagery may be composited at runtime to provide an unbounded base map. Robots avatars are placed in the scene based on the latest pose estimate and articulated by joint telemetry. An extensible library of visualizations is available for the user to display raw sensor data, derived data products, and other telemetry. For the purposes of the safe-guarded driving task, visualizations for navigation maps and terrain analysis trajectories were most commonly used.

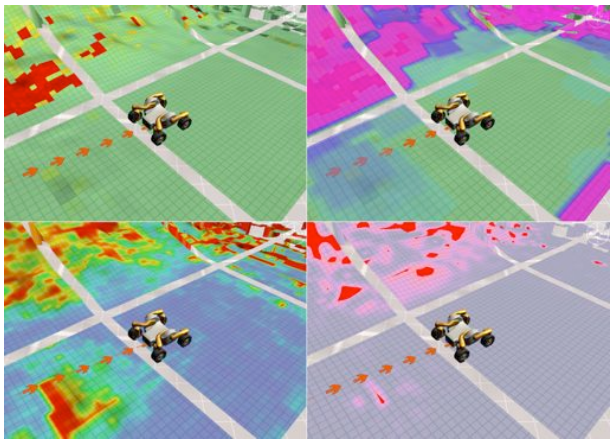

Figure 7: Navigation map rendering modes. Clockwise from upper left: traversability map, c-space map, slope map, roughness map

The RAPID navigation map data structure consists of a transform, a tile id, and an array of labeled layers which contain a grid of arbitrary data. Each layer is allowed

an independent subsampling factor to reduce map size on the wire. If a layer labeled as "height" is present in the map it will be used to construct the map tile mesh, otherwise, the mesh is constructed from either height values obtained from the base map, or from height layers obtained from another navigation map topic. Data from all other layers is uploaded directly to the GPU for processing. The user can choose how to display the navigation maps by selecting one of the GLSL shaders for different rendering modes, and widgets are provided to interactively modify shader parameters such as thresholds and color ramps. The operator typically uses the traversability shader while driving, although the roughness, slope, and c-space shaders (Figure 7) are helpful for gaining insight about the terrain when the safe-guarded driving module halts the vehicle.

The terrain analysis trajectory visualization gives the operator insight into how the safe-guarded driving module is perceiving the terrain. When a drive command is issued to the robot, the planned drive path is sampled at regular intervals and tested against the c-space navigation map to classify points along the path as safe, hazardous, or unknown. The result is visualized as a color coded trajectory with arrow icons placed at regular intervals to indicate orientation of the vehicle at along the path. Flashing icons indicate that the path has intersected a hazardous or unknown region within stopping distance of the vehicle.

6. RESULTS

In June 2012 we tested the tele-operation system at the JSC Rockyard (Figure 1). Four separate drives were accomplished by the operator (Figure 8) in a separate building, over a 1 Mbps bandwidth wireless link. At each command cycle the operator would review the maps and select the next waypoint. Multiple waypoints can be queued up, and execution may be interrupted at any time. An E-stop operator behind the robot was on hand to stop the vehicle if at any point the rover was in danger. No guidance was given to the driver by the E-stop operator.

Table 4 summarizes the drive results. No attempt has yet been made to systematically quantify the effects of changing system parameters (bandwidth, latency, detection sensitivity) on drive performance. Nonetheless we gained some qualitative insight to system performance. The one E-stop event occurred when the system was started up with a hazardous rock right in front of the vehicle, below the LIDAR field of view (the stereo hardware is in place to rectify this but is not yet tied in).

A demonstration highlight was summiting "Mt Kosmo", a 15m high hill with steep drop offs on 3 sides (Figure 9).The operator had to turn off the safe-guarded driving mode to get past some vegetation clumps that registered as obstacles (Figure 10, 11)

Craters (Figure 12) were hard to see in the camera images but showed up clearly in the maps (Figure 13). Crater interiors remained occluded until the rover was well inside (requiring the disabling of safeguarded driving).

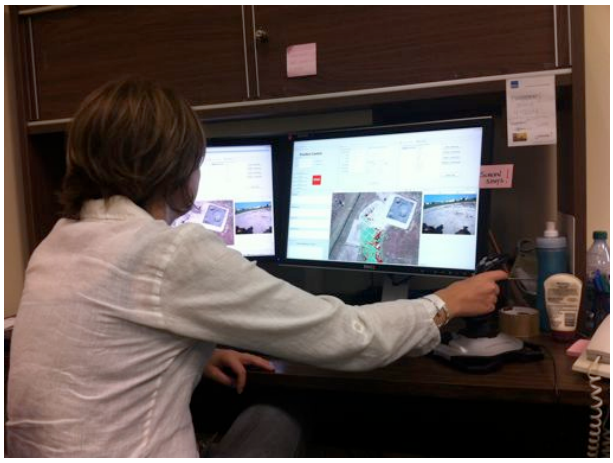

Figure 8: Centaur2 driver

Figure 9: Centaur2 at summit of Mt Kosmo

Hazardous rocks are detected (Figure 14), albeit doing so reliably currently requires turning off the slope compensation detailed earlier. Slope compensation is effective at reducing false positives but also reduces rock detection in dense rock fields (Figure 15).

Table 4: June 2012 tele-operation demonstration results

Test #	1	2	3	4
Distance	496m	435m	220 m	253m
# cmd cycles			13	10
Avg cycle dist			17m	25m
mean speed	0.24m/s	0.18 m/s	0.15m/s	0.14m/s
E-Stops	1			
False Stops			2	
True Stops				3

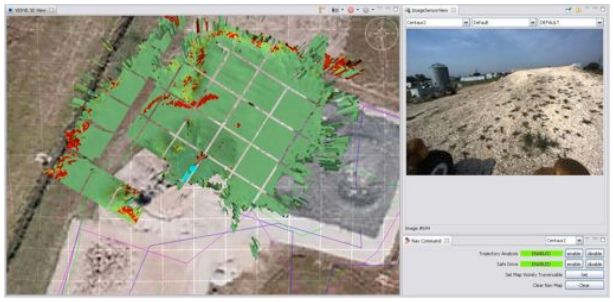

Figure 10: Navigation UI as Centaur2 ascended Mt Kosmo. Note the small false obstacles due to grass, which the operator over-rode by turning off the safe-guarded driving mode.

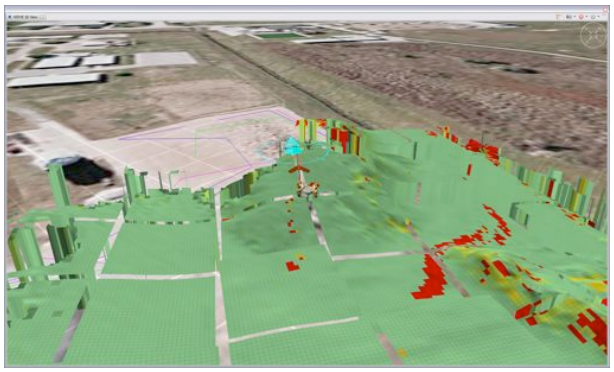

Figure 11: Oblique view of navigation map returned by Centaur2 as it ascended Mt Kosmo.

Figure 12: Centaur2 in crater.

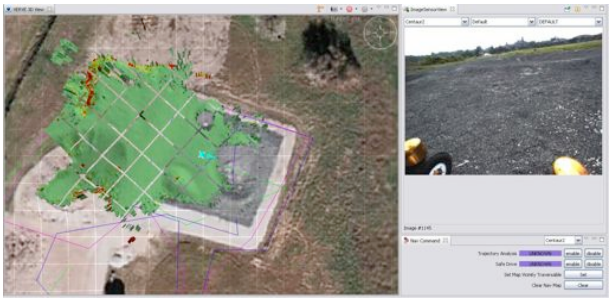

Figure 13: Driving around craters. Significant occluded areas inside craters resulted in path vetoes. Going inside requires the operator to override safe driving mode. Craters show up clearly on the maps, even when they are hard to see in the driving images.

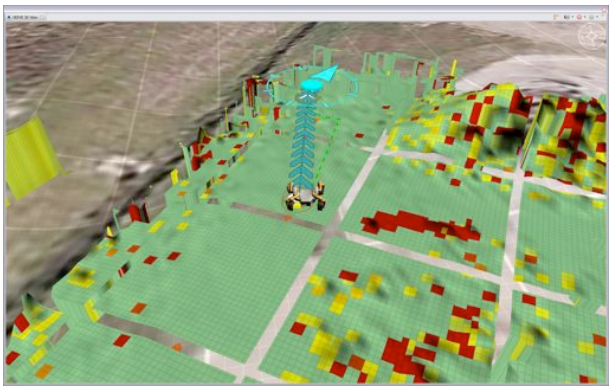

Figure 14: Rock field map with slope correction disabled, showing significantly better rock detection at the cost of greater sensitivity to vegetation on surrounding slopes.

Figure 15: Map of rock field (top) and E-Stop operator walking in front of rover (red trail) causing the robot to stop.

7. CONCLUSIONS

We have demonstrated a modular tele-operation system on two distinct NASA rovers with very different pose estimation systems. The novel map alignment algorithm is *essential* for the creation of consistent terrain maps, and enables the use of pose sensors of modest accuracy.

This project remains under active development at this time. Specific next steps include using the stereo vision system to detect hazards immediately in front of and below the rover (the LIDAR blind spot), visual odometry for improved local pose estimation, and aligning the LIDAR point clouds with the prior basemap DEM for a global pose correction that might eliminate the need for GPS entirely (if the basemap is of sufficient resolution).

ACKNOWLEDGMENTS

This work is funded by NASA's Human Robotics Systems (HRS) Project *Advanced Navigation Software* task, using the Centaur2 and SEV vehicles at NASA Johnson Space Center, and the KRex rover at NASA Ames Research Center.

REFERENCES

1. Beyer, R., Cockrell, J., Colaprete, A., Fong, T., Elphic, R., Heldmann, J., and Pedersen, L. Feasibility and definition of a lunar polar volatiles prospecting mission. In *Lunar and Planetary Science Conference*, 2011.

2. Seibert, M. personal communication, June 2012.

3. Pedersen, L., Han, C. S., and Vitus, M. Dark navigation : Sensing and rover navigation in permanently shadowed lunar craters. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS)*, 2008.

4. Bartlett, P., Wettergreen, D., and Whittaker, W. L. Design of the scarab rover for mobility and drilling in the lunar cold traps. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS)*, Los Angeles, California, February 2008.

5. Pedersen, L., Allan, M., To, V., Utz, H., Wojcikiewicz, W., and Chautems, C. High speed lunar navigation for crewed and remotely piloted vehicles. In *ISAIRAS*, Sapporo, Japan, September 2010.

6. Simmons, R., Henriksen, L., Chrisman, L., and Whelan, G. Obstacle avoidance and safeguarding for a lunar rover. In *AIAA Forum on Advanced Developments in Space Robotics*, Madison, WI, August 1996.

7. Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., and Mahoney, P. Winning the darpa grand challenge. *Journal of Field Robotics*, 2006.

8. Thrun, S., Montemerlo, M., and Aron, A. Probabilistic terrain analysis for high speed desert driving. In Sukhatme, G., Schaal, S., Burgard, W., and Fox, D., editors, *Robotics Science and Systems Conference*, Philadelphia, PA, 2006.

9. Torres, R., Allan, M., Hirsh, R., and Wallick, M. Rapid: Collaboration results from three nasa centers in commanding/monitoring lunar assets. In *Aerospace conference, IEEE*, pages 1–11, March 2009.

10. Object management group data distribution service (dds). `http://www.omg.org/spec/DDS`, 2012.

11. Flückiger, L. and Utz, H. Field tested service oriented robotic architecture: Case study. In *International Symposium on Artificial Intelligence, Robotics, and Automation in Space (iSAIRAS)*, 2012.

12. Robonaut2. `http://robonaut.jsc.nasa.gov/default.asp`, July 2011.

13. Krex rover development day. `https://ti.arc.nasa.gov/blog/irg/?p=485`, July 2012.

14. Low, K.-L. Linear least squares optimization for point-to-plane icp surface registration. Technical Report TR04-004, Department of Computer Science, University of North Carolina at Chapel Hill, February 2004.

15. Burridge, R. R. and Hambuchen, K. A. Using prediction to enhance remote robot supervision across time delay. In *IEEE/RSJ Intelligent Robotics and Systems (IROS)*, 2009.

16. Losasso, F. and Hoppe, H. Geometry clipmaps: Terrain rendering using nested regular grids. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 769–776. ACM, 2004.

17. Tanner, C., Migdal, C., and Jones, M. The clipmap: A virtual mipmap. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, pages 151–158. ACM, 1998.