HUMAN-ROBOT TEAMING STRATEGY FOR FAST TELEOPERATION OF A LUNAR RESOURCE EXPLORATION ROVER

International Symposium on Artificial Intelligence, Robotics and Automation in Space
Virtual Conference 19–23 October 2020

L-J. Burtz[1], F. Dubois[2], N. Guy[3]

[1]Amanogi Corp., Japan, E-mail: louis@amanogi.space

[2]Datamapab, Japan, E-mail: fabian@datamaplab.com

[3]Astroscale Japan Inc, Japan, E-mail: n.guy@astroscale.com

ABSTRACT

We leverage the Lunar South Pole resource exploration rover simulation environment from the NASA Space Robotics Challenge to propose a human-robot teaming strategy for fast teleoperation. We explore the applicability to mission scenarios by defining the roles and sharing of responsibilities between the human operator and the AI, implementing an AI running onboard the (simulated) rover, and implementing the command and data flow between the AI and the human operator through algorithmic explainability and a Graphical User Interface.

1 INTRODUCTION AND MISSION CONTEXT

This study focuses on the first few upcoming Lunar surface exploration missions. We are concerned with the missions launching in 2021 to 2025 that include a mobile rover that will prove the technology and lay the groundwork for enabling increased capability on each subsequent mission.

1.1 Human-Robot Teaming Rationale

In this context we prioritize robustness to unknown environments and a deployed navigation architecture that naturally evolves from iterative development. We make algorithmic explainability a first-class citizen to benefit development, commissioning on the Lunar surface, and fast identification and resolution of off-nominal scenarios.

This approach motivates a teaming strategy between the Artificial Intelligence onboard the rover and the human operator on console in Mission Control. The teaming leverages the advantages and mitigates the drawbacks of each actor:

AI onboard the rover:
- Advantages: access to high-framerate high-resolution data and no latency induced by Moon/Earth communications.
- Drawbacks: limited computing resources (mainly due to power and radiation constraints) and difficult to prove high reliability, especially in unknown environments.

Human operator in Mission Control:
- Advantages: human cognition and an appreciation of the entire mission context, future goals and past achievements. Holds accountability for mission success and is efficient at directing anomaly resolution teams.
- Drawbacks: prone to fatigue and some mistakes, requires repetitive tasks to be abstracted, only access to compressed, lower fidelity and delayed data.

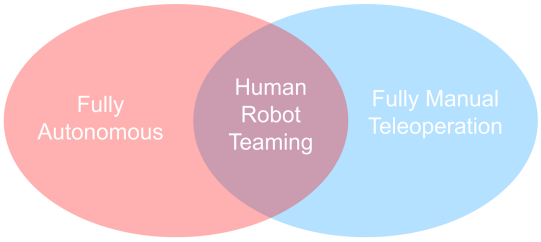

Figure 1: elementary human-robot teaming definition diagram

We define this human-robot teaming concept as a system whose components have the following characteristics:
- An AI running onboard the rover that can be fully autonomous in the majority of scenarios.
- An AI designed as a “white box” with explainability at its core, to enable real-time tele-monitoring by a human operator.
- An AI that is able to gracefully fallback to human tele-operation when encountering corner cases. This avoids compromising on AI capability and complexity while keeping development effort manageable.
- An operator that is trained to monitor the AI and manually perform the tasks the AI fails at.
- An operator within a team that includes the other rover operators, mission scientists and support engineers in Mission Control.

- A Graphical User Interface to intuitively monitor the AI and switch from "AI Behavior" to "Human Override."

1.2 Enablers of Human-Robot Teaming

This teaming strategy requires assumptions that we believe to be valid for the majority of the first Lunar surface mobile exploration missions.

First and foremost, this concept is only possible because of the relative proximity of the Moon to the Earth. This concept cannot be applied to Mars rover missions where the >6 min round trip communication time would be prohibitive. Second, the data link between the rover and Mission Control must be bi-directional, continuous, and reliable: subsequently referred to as Near Real Time (NRT) communication. This is either available as long as the rover is near the lander (e.g., ispace-inc Payload User Guide [14]) or as long as there is line of sight for direct-to-Earth communications (such as for the NASA VIPER rover [15]). These conditions will be met in the first phase of any mission and will most likely be met for the majority of the mission duration in the context of the first upcoming Lunar surface exploration missions.

A key enabler of this Near Real Time concept itself is that the bandwidth required is low (on the order of 100kbps downlink / 1kbps uplink), as we explain in greater detail in the section 4.1 Bandwidth Reduction.

Also critical to this concept are enablers that we will detail further in upcoming sections:

- Algorithmic explainability
- Clear separation of responsibilities between the human operator and the AI
- Graphical User Interfaces that are responsive, immersive and with a low psychological load to enable continuous long duration operation

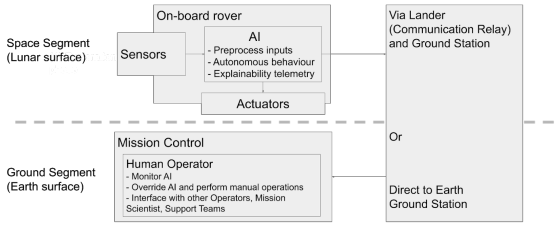

Figure 2: Overview functional diagram: Rover, Communications and Mission Control

1.3 Leveraging the NASA Space Robotics Challenge

We implemented this teaming strategy in the simulation environment of the NASA Space Robotics Challenge Phase 2 [2].

The competition provides a Lunar South Pole virtual environment implemented with the open-source Gazebo9 engine and ROS framework. Simulated four-wheeled rovers prospect for resources in the vicinity of the lander. Rovers are all equipped with a forward-facing stereo camera (40 cm baseline), LiDAR (150 degree wide field of view line scanning), tilt motor for the vision sensors, Inertial Measurement Unit (3-axis accelerometer and gyro), encoders on each wheel and steering arm (position and speed), and headlights. "Scout" type rovers are additionally equipped with a volatile sensor, analogous to a Neutron Spectrometer.

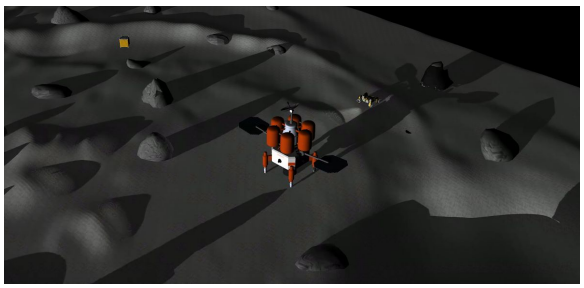

Figure 3: 3D simulation of the lunar surface with rover (yellow), lander (red) and probe (yellow cube)

Figure 3 shows the randomly generated Lunar terrain, with a lander and the scout rover that it delivered to the Lunar surface. The low-angle illumination is representative of polar illumination conditions. Five craters are marked as Permanently Shadowed Regions. The environment also includes 30 locations that are "volatile-rich" and that will trigger the volatile sensor within a 2m range. Finally, a cube is hidden within the environment, conceptually representing a lost probe (similar concept to Apollo 12 astronauts visiting Surveyor 3).

Within this environment, three tasks must be performed within 45 simulation minutes each:

- Task 1: using a scout rover, explore the environment to report the location of the volatile-rich regions. This task requires accurate localization combined with an efficient exploration strategy.
- Task 2: using two rovers (an excavator rover and a hauler rover), navigate to volatile-rich areas to excavate regolith and drop it into the hauler's bin. This task requires accurate localization and precise coordination between two rovers.
- Task 3: using a scout rover, find a lost probe hidden in the environment and then return to the lander (with a specific parking pose relative to the lander) as quickly as possible. This task requires an efficient exploration strategy and precise object recognition and relative pose estimation.

We leverage the NASA competition for the simulation environment as well as the relevance of

the tasks and rover concepts to upcoming missions. Task 1 and 3 are particularly relevant to the first prospecting missions and will therefore be the main focus of this study. For the NASA competition itself, as per the rules, we submitted a fully autonomous AI with no human input. In this study, however, we explain how we took the concept further by:

- Exposing the "thought process" of the AI (inputs, intermediate computations, and outputs) in a Graphical User Interface to the human operator
- Adding the capability for the human operator to send direct commands to the rover or modify higher level goals for the AI to follow
- Adding an interface behavior to allow the human operator to override the AI, perform manual actions and hand over control back to the AI

1.4 Outline

The next sections present the teaming strategy through a design reference scenario (section 2), specific proofs of concept (section 3) and discussion on the path to mission implementation (section 4).

Since this human-robot teaming concept is inherently dynamic, videos are available in a dedicated public repository (https://bit.ly/2SBS7Cy).

2 DESIGN REFERENCE SCENARIO WITH HUMAN-ROBOT TEAMING

2.1 Separation of Responsibilities and Overview of the Design Reference Scenario

To better understand the robot-human teaming, we present a Design Reference Scenario (based on the Task 3 of the NASA competition). This task requires the rover to leave the vicinity of the lander, explore the environment to find and approach a lost probe, precisely compute the relative distance between the rover and the probe, then return to the lander and precisely align with a fiducial marker on one side of the lander (Figure 4).

Figure 5 presents the timeline of this task, highlighting the separation of responsibilities between the AI and the human operator as the task progresses. In parallel, at any time that the rover is moving, the AI behavior and human operator have the responsibilities outlined in Figure 6.

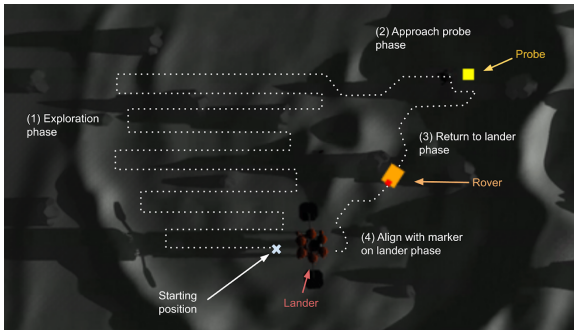

Figure 4: Rover's path during the four phases of the Design Reference Scenario

2.2 Use of Behavior Trees

While Finite State Machines (FSMs) have a long history of being used in robotics, their main drawback is their lack of reactivity and modularity. Behavior Trees (BTs) solve these two issues using two-way control transfer instead of one-way control transfer [3].

A behavior tree can be represented as a tree structure. Leaves are either conditions or action nodes. Other nodes of the tree are control flow nodes. A BT can itself take the role of an action in another BT, contributing to modularity.

Execution of the BT occurs at a fixed time interval, where a tick signal is generated and propagated from parent node to child node according to the control flow rules. A node can return three execution statuses: *success*, *failure*, and *running*. Execution finishes when the root node returns its execution status.

The Sequence control flow node *(*symbolized by ->) executes all child nodes until one node returns *failure* or *running*. If all nodes succeed, it returns *success*. The Fallback (also called *Selector*) control flow node *(*symbolized by *?*) executes the child nodes until one node returns *success* or *running*. If all nodes fail, it returns *failure*

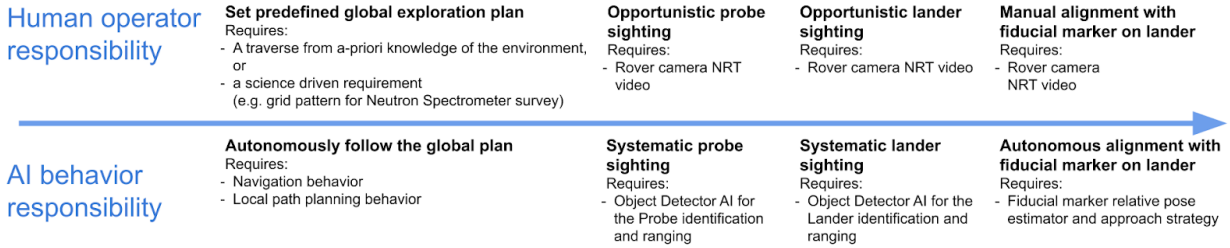

Figure 5: Timeline of general tasks that are either the responsibility of the human operator or the AI

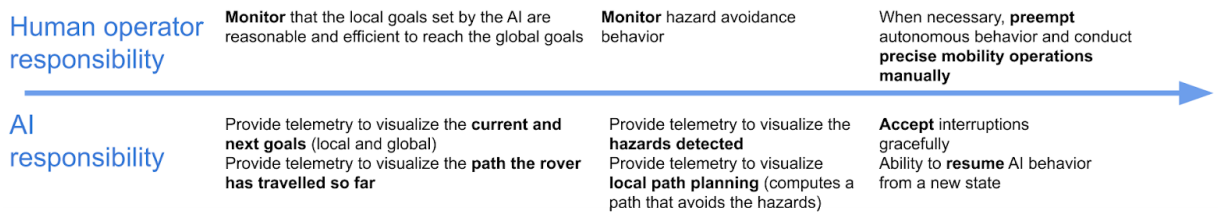

Figure 6: Timeline of recurrent mobility tasks that are either the responsibility of the human operator or the AI

2.3 Behavior Tree Implementation of AI

We use a minimal implementation using only Sequences and Fallback control nodes with the possible use of state variables. This is sufficient to generalize decision trees, finite state machines and teleo-reactive approaches [3].

To improve reactivity, we try to use stateless idempotent tasks (e.g., turnHeadlightsOn). However, it is sometimes necessary to use nodes with memory (state) to keep idempotence and prevent an action from being executed repeatedly (e.g., "move 1 meter forward" being retriggered at each tick, leading to a never-ending traverse).

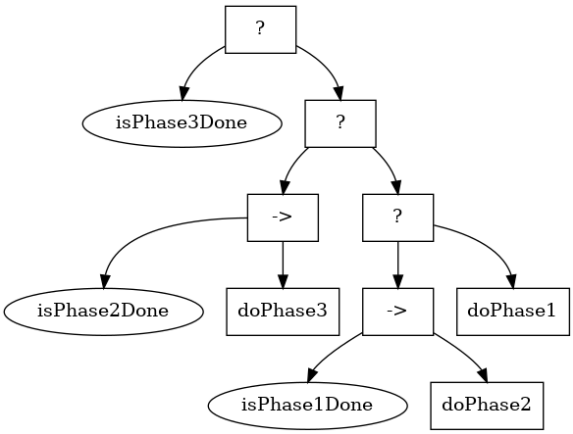

Figure 7: High level behavior for the Design Reference Scenario (NASA competition round 3)

For the mission itself, we achieve a goal-oriented design, using the fact that BTs generalize teleo-reactive approaches. Implicit sequences make the design a succession of goals (post conditions) and tasks required to achieve each goal, along with their preconditions. This is visible in this high-level tree for the Design Reference Scenario (Fig 7) where three phases have been identified.

In contrast with the overall behavior, which prevents the re-execution of an achieved goal, the subtree for Phase 1 (Fig 8) will re-execute former tasks if a precondition is no longer true. For example, losing sight of the probe will retrigger the search behavior. This is an example of reactivity creating more robust behaviors.

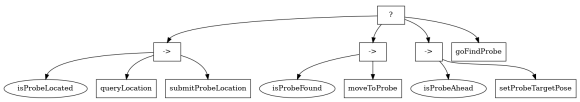

Figure 8: Behavior for the probe search phase doPhase1 (simplified)

2.4 Teaming with Behavior Trees

Since BTs generalize decision trees, we can create an auto/manual handover mechanism.

Fig. 9 shows the overall BT, where the autoMode node refers to the fully autonomous BT for the mission in progress (for example Fig 7). We see that the modularity of BTs is beneficial, since the details of the autonomous behavior can be ignored at this level.

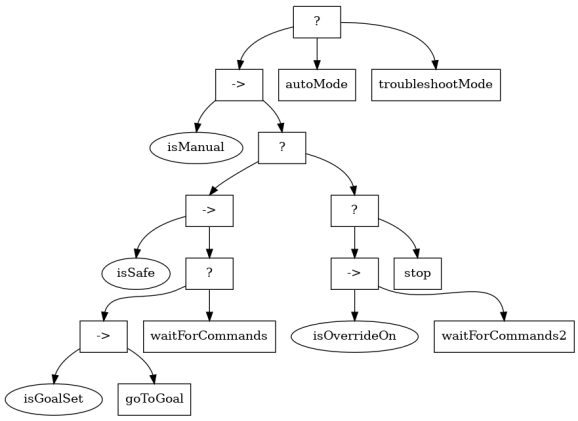

Figure 9: Human operator / AI behavior control switching implementation

Two levels of handover are available. At a high level, the manual/autonomous switch transfers mobility control between the AI and the human operator. This allows the operator to either send high-level goal coordinates to the planner via goToGoal, or direct low-level commands via waitForCommands (*go straight*, *turn*, *stop*).

In both manual and automatic modes, by default a safety check of the absence of obstacles in close proximity to the rover (called "Auto Emergency Stop") is active. This low level safety check can also be disabled by the human operator.

2.5 Navigation Teaming via Factor Graphs

The navigation capability is one of the main system design drivers for a Lunar rover. Localization and mapping are needed to ensure safety of the rover (not losing communication coverage and not going into hazardous regions) and for mission objectives (report where interesting instrument observations have been made and follow a systematic survey strategy). The teaming concept allows us to select a specific navigation strategy that leverages:

- access to high-resolution/high-frame rate sensor input onboard the rover (but this is limited by computational power)
- access to the human cognition of the operator in Mission Control

We propose a teaming strategy for Simultaneous Localization and Mapping (SLAM) implemented via a factor graph. This graph links together observations (factors) at high semantic level (probe, lander, landmarks sightings) completed by odometry at high frequency. The resulting graph is then solved with Multi-modal iSAM [7]. The factor graph contains variables representing the pose of the rover or landmarks at different points in time, and related to each other by factors representing knowledge about absolute or relative positions. For example, with odometry, we can relate to rover poses in the graph using the pose delta in the odometry frame. Observations of landmarks such as the lander using the camera also create a factor between the rover pose and the lander at observation time.

We choose a high semantic level implementation because of the lack of features in the Lunar environment (both in the simulation and in difficult illumination conditions on the Moon), to reduce the size of the graph (lower computational cost for solving and lower bandwidth cost for downlink) and to make the graph more human interpretable.

We provide interfaces for the human operator to interact with the factor graph in multiple ways:

- Annotation of observations: by tagging or image annotations
- Removal of observations: by reviewing sensor inputs at and around the time a factor has been added. This can be used to remove erroneous or imprecise matches.
- Addition of observations**:** to add observations that were missed or to add external observations (e.g., observation from an orbiter, or from a camera on the lander) or new types of landmarks.

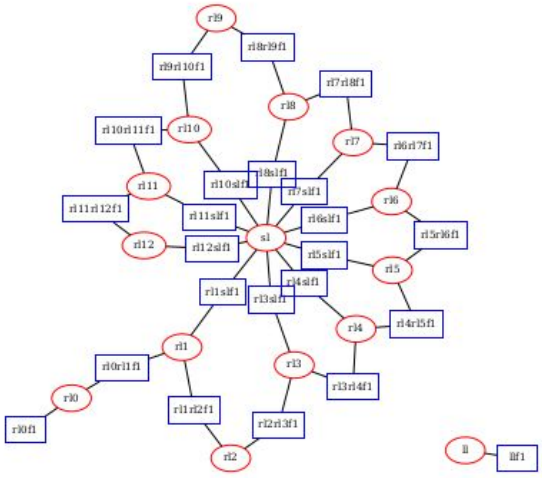

Figure 10: A factor graph after a few observations (blue rectangular nodes) linking rover poses and landmarks (red nodes)

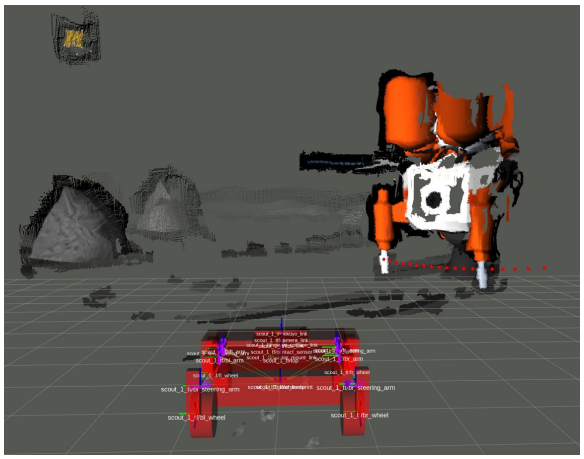

Figure 11: Visualization of the point cloud from the stereo camera (with visualization of the rover body for context), to manually add or review landmarks

3 SHOWCASES: SPECIFIC PROOFS OF CONCEPT

3.1 Overcoming Hazardous Terrain

Obstacle detection itself uses input from a horizontal LiDAR line scan, which is processed into higher level primitives: segments and circles representing discrete obstacles (rocks, slopes, the lander, etc.). The rover's local path planning uses the Timed Elastic Band (TEB) algorithm [8] to compute the optimal path toward a given goal that avoids obstacles. For safety, the hazard avoidance settings are deliberately conservative, and sometimes result in the rover going around hazardous terrain (e.g., a medium slope hill) when it would in fact have been safe to go straight through the hazardous terrain (e.g., over the hill).

The human operator can assess the situation more completely (using more context: camera image, previous experience, etc.) and decide that the most efficient trajectory (including safety) to reach the next goal is actually through the hazardous terrain.
The human operator therefore sends the more efficient commands manually [5], before handing over back to the AI that will resume execution within the new context (e.g., after the hill).

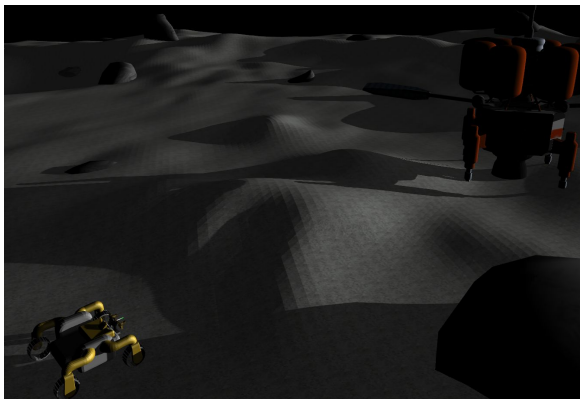

Figure 12: Rover in front of a hill blocking it from its goal (in this case the lander). Human operator experience can determine it is safe to go over this hill

3.2 Call for Assistance

The AI has the capability to detect that it is confronted to a new or dangerous situation. In these cases, the behavior is set to stop the rover and switch to human control. Once the problem is identified and solved manually, the solution can be encoded into a new behavior (effectively teaching the AI).

3.3 Precision and Context-aware Mobility

At the end of the Design Reference Scenario (Task 3 of the NASA challenge), the last task is to find the fiducial marker on the lander and precisely align with it (this task conceptually represents connecting to a charging port or handing over samples to the lander for processing).

The AI behavior uses as input a Marker Pose node that matches the features of a known model of the fiducial marker with the features in the rover's camera image (see Appendix 1 Technology Stack) to determine the relative pose (position and orientation) of the rover to the fiducial marker (Figure 13). However, in the presence of noise in the camera (simulated ionizing radiation effects) and difficult illumination conditions (e.g shadow of the lander) the AI behavior sometimes struggles to perform the precise alignment.

In these situations, the human operator takes over and, using the front camera as input, sends the precise mobility commands needed for final positioning and alignment. This concept is similar in essence to the manual grappling of some visiting vehicles to the International Space Station by astronauts operating the CanadaArm (e.g SpaceX Cargo Dragon, JAXA HTV).

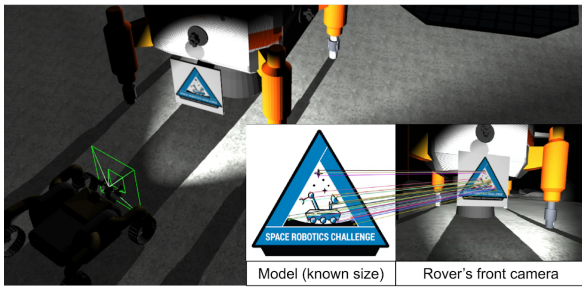

Figure 13: Rover aligning with fiducial marker. (inset right) feature descriptor matching for 3D/2D correspondence and camera pose computation

3.4 Opportunistic Detections and Science

Regarding the probe detection task, the operator can sometimes detect the probe (or visual cues that lead to the probe) before the AI. In these cases, the operator can switch to "Human Override" mode and set a goal toward the probe. Once the automatic detection of the probe becomes stable, the operator can give control back to the AI. The reactiveness of Behavior Trees to new inputs means that the AI won't resume at the exploration task but will resume directly at the probe approach task.

This concept is also applied for opportunistic science where a human operator (e.g. mission scientist) identifies an object of interest visually or through a combination of sensor readings (e.g from the Neutron Spectrometer). In this case, the human operator preempts the AI behavior and selects a new goal toward that object of interest. The object can also be added to the factor graph. This use case is inspired by the opportunistic science behaviors on the Mars rovers [13].

4 DISCUSSION: TOWARD FLIGHT IMPLEMENTATION

We considered the relevance of the proposed human-robot strategy to upcoming Lunar exploration missions from the beginning of the design phase. In this section, we highlight some of the main considerations in the design space: bandwidth reduction, mission profiles and mission phases.

4.1 Bandwidth Reduction

While we did not simulate communication delays, Quality of Service and bandwidth limitation during this study, these considerations were included in our design approach. Specifically, the visualizations

needed for our teaming strategy rely only on high semantic level, low bandwidth telemetry. The processes onboard the rover reduce the needed bandwidth in two ways:

1. Traditional compression of images / LiDAR point cloud, stereo camera point cloud.
2. Increasing levels of abstraction in the AI behavior we implemented naturally lead to low data rate primitives.

Regarding the first point: The only large data needed to be downlinked in real time to support the teaming strategy is one NRT feed from one camera. At 2 fps, 640x480 resolution, h264 hardware compression, this requires only about 50kbps [12].

The remaining large data is stored onboard the rover and will be downlinked during non-driving phases (pauses for battery charging, thermal balance and operator shift changes). While these raw datasets are not used for the real-time teaming strategy, they are valuable for the science and engineering output of the mission, and for better long term planning during the mission.

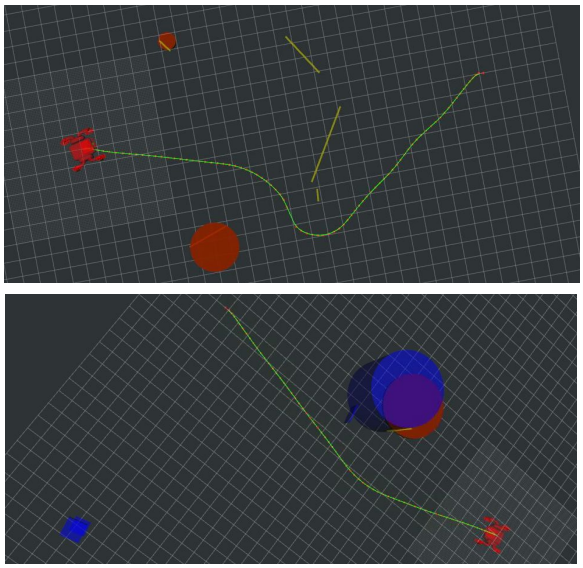

Figure 14: 3D visualization of the low-bandwidth primitives: (top) local path (green) between rocks (red disks) and slope (yellow segments) (bottom) lander coordinates (blue cylinder) and probe coordinates (blue cube) from Vision AI

The second point is more interesting: the remaining data needed for the teaming strategy consists of high-level, discrete algorithm outputs:

- Circle center coordinate and radius for each obstacle of the LiDAR instead of a full line scan.
- 3D coordinates of the position of the lander, rover, marker and probe (only when detected) taken from the stereo point cloud instead of the full point cloud.
- Behavior logic and states (several hundred bytes only)

The reliance on pre-processing and an active AI onboard the rover allows for significant bandwidth reduction during the active phases (rover moving). This concept is a significant enabler considering the effort and cost for the infrastructure needed to support Lunar communications.

4.2 Mission Profiles

The teaming strategy is relevant to several mission profiles:

- Technology demonstration mission: exploration around the lander
- Traverse to and between science stations (eg NASA VIPER concept of operations)
- Science payload (e.g. GPR or Neutron spectrometer) need to follow a specific grid pattern for a systematic survey.
- Virtually all mission profiles that stay within communication range of the lander (Payload User Guides of CLPS lander providers show that communication range can be expected to be within 250m/500m of the lander) or mission profiles that include direct-to-Earth rover communications (e.g. the NASA VIPER rover)

For any mission profiles, the teaming strategy is particularly relevant:

1. At the start of the mission for efficient commissioning activities
2. During off-nominal scenarios for performing Failure Detection Isolation and Recovery (FDIR)

First, the start of any mission carries significant risk and lessons learned show that the shakeout of all systems to verify that they are functioning as designed yielded surprises. Particularly for the first few upcoming lunar surface exploration missions, we expect surprises related to navigation tasks due to:

- difference between the simulated environment and lunar analogs used for development (illumination conditions, hazard distribution),
- difference between the planned and the actual landing site, and
- issues in the performance of some rover subsystems (issues in any of vision, mobility, power, thermal or communication subsystems could require adopting a new exploration strategy).

The teaming strategy proposed, specifically the ease of modifying and adding behaviors online during the first hours of the mission would prove invaluable.

Second, system engineering best practices consider off-nominal scenarios from the design phase. The

teaming strategy provides inherent explainability and ability to monitor the inner workings of the AI behavior. This capability creates a constant failure detection stance and provides dedicated, well rehearsed pathways to failure isolation and recover. Keeping the human in the loop also makes for more flexible reconfiguration and adaptation to degraded performance (as long as the communication link is available).

4.3 Conclusion

The specifics of the next few upcoming lunar surface exploration missions motivate a human-robot teaming strategy to achieve fast teleoperation of rovers.

The teaming strategy is particularly well suited in the context of exploration (uncertain, changing, complex goals), high capability, high reliability, manageable development effort.

The teaming strategy is a stepping stone towards fully autonomous missions by monitoring autonomous behavior in-situ and collecting the treasure trove of data to properly inform the design of a fully autonomous concept of operations for the subsequent mission. The teaming strategy makes the most of the human operator's capabilities in the first stages of a mission while a communication link is available and prepares the rover in the best possible way for the more ambitious beyond communication range phases.

The teaming strategy makes for efficient iterative development which naturally evolves into the flight implementation.

Fully manual teleoperation is too slow for short mission durations (most concepts are less than 14 Earth days) and does not scale well to multiple rovers and the needed mission complexity. Fully autonomous concepts of operations risk delaying the first mission due to the complexities of development, testing and mission assurance.

Acknowledgement

We express our sincere gratitude to the NASA Space Robotics Challenge team and sponsors for the simulation environment and encouragement to publish results that expand on the competition itself.
We thank the authors of open source software and the ROS community that made rapid iterative development of the concepts presented in this study (and our submission to the NASA competition) possible. Direct links to their code repository or reference are in Appendix 1 Technology Stack.

References

[1] Gaines D. et al. (2016) *Productivity Challenges for Mars Rover Operations: A Case Study of Mars Science Laboratory Operations.* AAAI Int. Conf. Automated Planning and Scheduling

[2] NASA Space Robotics Challenge Phase 2 www.nasa.gov/directorates/spacetech/centennial_challenges/space_robotics/about.html (Last Updated: Aug.12, 2019)

[3] Colledanchise M. and Ögren P. *Behavior Trees in Robotics and AI: An Introduction.* (2017) arXiv preprint, arXiv:1709.00084.

[4] Côté N. et al. *Humans-Robots Sliding Collaboration Control in Complex Environments with Adjustable Autonomy.* (2012) IEEE/WIC/ACM.

[5] Cognetti, M. et al. *Perception-Aware Human-Assisted Navigation of Mobile Robots on Persistent Trajectories.* (2020) IEEE/Robotics and Automation Letters.

[6] Nashed, S.B. and Biswas, J. *Human-in-the-Loop SLAM.* (2018) 32nd AAAI Conference on Artificial Intelligence.

[7] Fourie D., et al. (2016) *A Nonparametric Belief Solution to the Bayes Tree*. IROS

[8] Oriol Gasquez, (2018) *Hakuto Flight Model User Interface.* Personal website, while working at ispace-inc, with permission: https://oriol.gasquez.com

[9] C. Rösmann, F. Hoffmann and T. Bertram, (2017) *Integrated online trajectory planning and optimization in distinctive topologies.* Robotics and Autonomous Systems, Vol. 88, 2017, pp. 142–153.

[10] Uno, K., Burtz, L.-J., Hulcelle, M. & Yoshida, K. (2018) *Qualification of a Time-of-Flight Camera as a Hazard Detection and Avoidance Sensor for a Moon Exploration Microrover*. Transactions of the Japan Society for Aeronautical and Space Sciences

[11] Moore, T., Stouch, D. (2016). *A generalized extended kalman filter implementation for the robot operating system*. In Intelligent autonomous systems 13 (pp. 335-348).

[12] Walker, J. (2018) *Flight System Architecture of the Sorato Lunar Rover.* 16th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), Madrid, Spain.

[13] Castano, R., et al. (2007). *Oasis: Onboard autonomous science investigation system for opportunistic rover science*. *Journal of Field Robotics, 24(5), 379-397.*

[14] ispace-inc team. *(2020) Payload User's Guide v2.0,* Available at https://ispace-inc.com/wp-content/uploads/2020/05/ispace_PayladUserGuide_v2_20200 1.pdf, Accessed Oct. 5th, 2020.

[15] Ennico-Smith, et al. (2020). *The Volatiles Investigating Polar Exploration Rover Payload.* Lunar Planetary Institute, (2326), 2898.

APPENDIX 1: TECHNOLOGY STACK

Function	Data input	Methodology	Reference
Level 1			
Hazard Detection	LiDAR (front-facing 2D line scan)	Merge returns into segments. Merge short segments into circular obstacles	Based on obstacle_detector ROS package
Rover Odometry	Wheel and steering arm encoders + IMU (3-axis accelerometer and gyro)	Fusion via Extended Kalman Filter	Based on robot_localization ROS package [11]
Local path planning	Current position + next local goal ~5 to 25m away + Hazards detected	Compute intermediate poses to reach the next goal while avoiding hazards and optimizing for shortest distance/time	Based on teb_local_planner ROS package [9]
Stereo Ranging	Time synchronized image pair from front stereo camera + camera calibration	Compute disparity and point cloud with Block-Matching algorithm	Based on stereo_image_proc
Object Detection	Monocular camera image	Compute bounding boxes for the Lander / Marker / Probe / other Rover	Custom detector based on OpenCV color extraction in HSV color space and morphological operations
Level 2			
Vision AI	Point Cloud from Stereo processing Bounding box from Object Detection	Matching, filtering and frame transformations to output relative position between object and rover	Custom heuristic for combining inputs, rejecting noise, and providing a robust output
Marker Pose for precise alignment of rover with lander	Time-synchronized image pair from the front stereo camera + bounding box of marker from Object Detector	Precisely compute the relative camera pose (position + orientation) to the fiducial marker on the lander	Based on OpenCV implementation of ORB feature extraction and description, brute force matching, and solving PnP 2D/3D correspondence with RANSAC
Level 3			
Autonomous behavior and decision making	Rover Odometry Local path planning Vision AI, MarkerPose Volatile sensor human operator input	Behavior Tree for teleo-reactive approach Modular re-usable and mostly stateless behaviors see Section 2.3	Custom behavior tree library
Navigation / Simultaneous Localization and Mapping	Rover Odometry Vision AI human operator input	Limited number of factors but high quality and high semantic level. Unified framework for including disparate factors (odom/visual ranging/human labels/outside knowledge)	Based on Multi-modal RoME.jl implementation of iSAM [7] in Julia
Infrastructure			
Simulation environment and physics	Models and plugins by NASA SRP2	Gazebo9 engine, physics by Open Dynamics Engine	All credit goes to the team at NASA SRCP2
Middleware	N/A	Built from source with Python 3	ROS Melodic
Front End	RVIZ for the 3D UI ipywidgets for the interactive dashboard UI plot_juggler for all time series plotting (most valuable tool for development and troubleshooting)		