Space Robotics Challenge Phase 2
Electronic Summary
Team Robotika

Frantisek Brabec, PhD
Annapolis, MD
brabec@gmail.com
frantisek.brabec@robotika-intl.com
www.robotika-intl.com

I. INTRODUCTION

Our solution is built around two independent teams of robots, each containing one robot of each type (i.e., scout, excavator and hauler). Each team is assigned half the workspace (a semicircle) to explore and excavate. A small buffer between the two semicircles is maintained to prevent the teams from interfering with each other.

We have contemplated other team compositions, such as a single scout, two excavators and three haulers - that way further excavation wouldn't have to wait for a hauler to unload previously dug up volatiles first since another hauler would be available to take over immediately. However, it was not clear that the haulers' delivery capacity was indeed the bottleneck, many times the hauler is able to go to the processing plant and onto the next volatile quicker than the much slower excavator traveling to the next volatile directly. Even more important was for us the higher robustness of the solution based on two scouts. With only one scout it becomes the weakest link of the solution and possibly a bottleneck as well. If something were to go wrong during the scout exploration, with two scouts the overall operations can still continue even though with degraded capacity. Finally, we observed that maintaining the rover's current location was not maintained accurately enough to allow a scout to report coordinates of a volatile and immediately continue exploring expecting the excavator and hauler to arrive at the same destination later based on the coordinates alone.

Hence, the approach we have chosen is for the scout to explore the workspace in a systematic manner and when it locates a volatile, it centers over it and invites its teammates (excavator and hauler) to approach. They arrive to the general vicinity based on the coordinates provided by the scout and finish the approach visually for maximum accuracy. Once the excavator reaches the scout, the scout continues its search for another volatile while the excavator and hauler proceed to dig the volatile up and hauler then delivers it to the processing plant. The process then repeats.

Any time any rover is not performing a specific task, it turns itself towards the sun in order to charge its batteries. If a rover's battery runs low while on a move, a detour is performed towards the recharging station. Then, once the battery is full, the rover resumes its latest task. If for some reason the battery is depleted before the rover reaches the recharging station, it stops and charges in place through its solar panels to increase its battery charge by a few percentage points, then it continues towards the recharging plant for full recharge. We leave enough battery capacity in reserve so that a rover cannot run out of energy while performing a non-move task (e.g., digging).

Towards the end of the run, about 3 sim-minutes before the cutoff time (i.e., 2 simulated hours), if digging and loading is in progress at either team, that loading process is interrupted and the hauler proceeds towards the processing plant to unload any volatiles it may have in the bin at that time to score points for them as well.

Given that the solution grading is based on the average score across three runs (as opposed to the best-out-of-three score), it is clear that reliability of the solution is one of its most important properties of a competitive solution, even at the cost of introducing inefficiencies. We felt that being able to maintain a working solution for six simulated hours will lead to a better outcome than adopting a more efficient but even just slightly riskier approach. Hence, our strategy prioritizes methods that yield a more robust solution. This typically involved decisions around parallel operations, robot velocities, obstacle handling or maintaining separation between robots.

II. METHODS

Building blocks

Our overall solution utilizes the following components.

Sensor filtering

Three types of ROS topics need to be filtered to minimize the amount of noise that the solution then works with. Utilizing noisy input directly could lead to less robust localization, object detection and obstacle avoidance. Each filter takes in the raw sensor input and publishes the same type of data as its output under a different topic name (e.g., *imu* to *imu_filtered*). The following topics are filtered:

1. IMU - the IMU filter calculates a weighted average of the last 10 readings (with more weight towards the more recent ones)
2. Camera images - the camera filter takes a narrow strip of pixels along the left edge of the image starting at the top (50px wide, 300px high). It then performs a blurring function on it and then compares the standard deviation between the original and blurred image segment. The idea is that a non-noisy frame (especially in that part of the view) typically has very monotonous content (black sky) while a noisy image has much more deviation. If the difference between the standard deviations in blurred and non-blurred versions of the image fragment differ by more than a certain threshold, the image is identified as noisy and is skipped from republishing into the output topic (at most three images are skipped in a row). While this method doesn't remove 100% of noisy frames (at least the way we tuned its parameters), it works well enough for image noise not to be a problem for us in the rest of the solution.
3. Laser - the laser filter first calculates a median value for each three consecutive rays, then it takes the average of the current value and the value from the previous most

recent scan from the same ray. The resulting value for each ray filtered this way is published into the output topic.

Swerve driving

The solution controls the robots' driving using the standard ROS Twist messages. Swerve driving module converts Twist into steering and velocity commands for the individual wheels. It is based on a Pull Request[1] to the standard ROS ros-controllers library. We have added an emergency mode where the full desired velocity is applied to the wheels immediately without honoring maximum acceleration/deceleration parameters. This is used for example if a robot needs to immediately start driving in the direction of the downhill slope to avoid flipping over. We have also added an option not to apply velocity to any of the wheels until for all of them is no more than π/4 difference between the current and desired steering angle.

The same module also provides the reverse service: it calculates odometry information for the overall robot from steering angles and velocities of all wheels as provided by the wheels' encoders.

Visual odometry

We are using RTabMap ROS module[2] to derive odometry information from the stereo camera topics.

Robot localization

We are using the ROS robot_localization module[3] to fuse sensor data together into odometry information. Specifically, we are fusing odometry from the visual odometry module, odometry from the swerve driving module and data from the IMU. We operate this module in 3D mode. IMU is the only data source used to maintain yaw of the robot.

Object detection

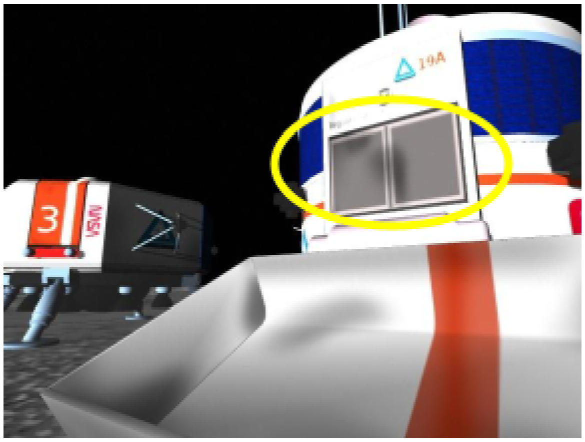
Our solution depends heavily on object detection from within the robot camera streams. We are using YOLOv3[4] to detect the following objects:

1. Rover (we do not attempt to differentiate between individual types of robots)
2. Processing plant
3. Repair station
4. Arm (on the excavator)[5]

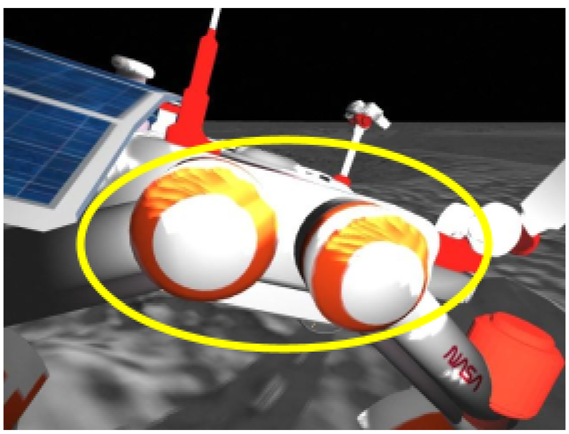
---

[1] https://github.com/ros-controls/ros_controllers/pull/441
[2] http://wiki.ros.org/rtabmap_ros
[3] http://wiki.ros.org/robot_localization
[4] https://pjreddie.com/darknet/yolo/
[5] This object type ended up not being utilized

5. The large gray rectangle above the processing plant hopper



6. The joints in the center of each side of a rover



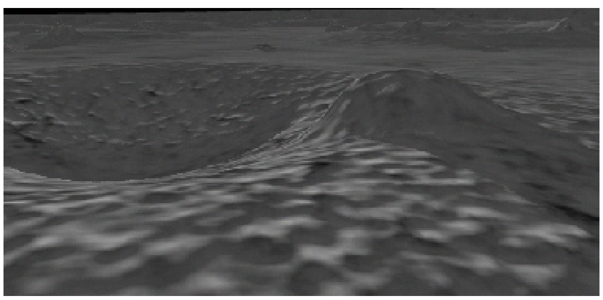
We have created a dataset of about 1,300 images to train the YOLOv3 module on. A single instance of the YOLO module is used to provide the detection service for all six rovers with frequency of 10Hz (in sim time). The detections are reported via a topic[6] containing the list of objects detected in the latest image, each specifying the object type and its bounding box.

Terrain scanning

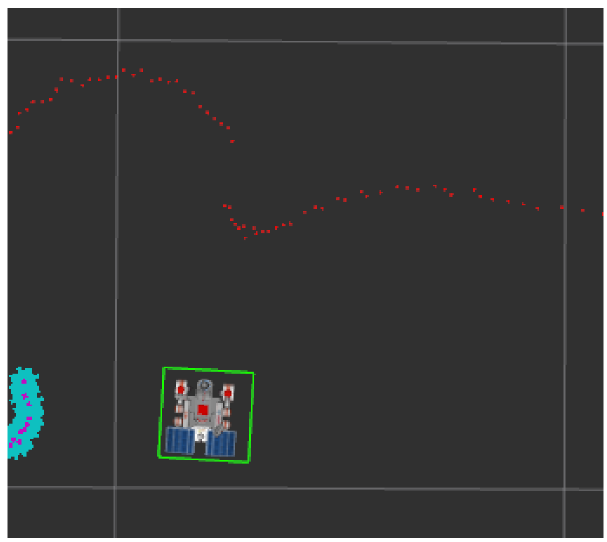
While the rovers can often successfully drive over the terrain without paying attention to any potential obstacles such as craters or boulders, this cannot be relied on for the whole 2-hour run. Without avoiding such trickier situations the rover will soon start losing locational accuracy due to abrupt movements as it bounces over obstacles and would eventually even flip over in a particularly unfortunate terrain configuration. As such, it is necessary to detect not just man-made objects such as other rovers or the landers but also the boulders embedded in the surface.

[6] https://github.com/frantisekbrabec/darknet_ros/tree/moon-single-node

To that end, we are dynamically tilting the laser such that its readings maintain a median distance of 4m. As the sensor bar tilt changes very slowly, if the median distance does not fall within the desired range around 4m, the rover slows down substantially until the tilt is adjusted such that the distance criteria is met.
As a result of the tilt, the curve formed by the individual laser hits on a flat surface is a straight line. When driving over uneven terrain, the shape of the laser curve becomes more complex. Our solution detects differences of this curve from a straight line and places obstacles into the rover's local obstacle map accordingly.
Due to the nature of this detection process, obstacles are only detected by the laser scan briefly as the laser beams sweep over them while at a given distance, if the obstacle is closer or further, it is not detected by this method. Thus we cannot remove an obstacle from the map as soon as it is no longer detected. Instead, we leave it in the map for approximately 20 seconds at which point it is purged automatically.



Crater to the left is combined with a boulder on its rim to the right



The same situation as captured by the lidar sensor. The protrusion in the middle pointing towards the rover is identified as an obstacle.

Mapping and obstacle avoidance

Our solution doesn't build or maintain a global map. It only maintains a local map of size 12x12m around each rover (maps for individual rovers are independent and do not interact with each other). Any rover movement (travel to a certain coordinate location or visual approach towards an object) can optionally be subject to limitations imposed by the obstacles represented in the local map.

Obstacles can be inserted into the map either by the terrain scanning module (see Terrain scanning) or by the object detecting module. If an object is detected and its estimated distance falls within the local map, it is added into the map. The object is cleared automatically after 3 seconds (not immediately, to accommodate brief losses in object detection due to temporary obstructions of the view or driving through craters, etc). Of course, if the object remains visible, it will be added to the map again and hence it maintains continuous presence in the map despite the automatic removal.

When an obstacle is detected on the path the rover would travel on within the next 3 seconds given the current Twist command, an alternate Twist command is calculated such that it avoids this (and any other) obstacle and is closest to the originally planned one.

For coordinate location based driving the avoidance is done by driving sideways (i.e. with angular velocity zero and thus while maintaining the rover's yaw). In the case of a visual approach, the angular velocity (angular z) which is calculated such that the rover keeps the object being approached within the view is maintained and only forward (linear x) and sideway (linear y) velocities are affected.

If no clear path seemingly exists in any direction or a rover is indicated as already overlapping an obstacle, the rover stops first, then drives backward before attempting to resume the original travel plan. Note that the obstacles introduced by the Terrain scanning module eventually expire automatically. Therefore, even if the terrain surrounding the rover is such that passage seems impossible in all directions, objects in the areas that are no longer being scanned by the laser will eventually be removed from the map and the rover will feel free to drive further in those directions. While this poses some risk as there could have been true unpassable obstacles, we rely on our other driving modules (see Emergency recovery) for enabling the rover to extricate itself from this location and continuing with its planned route.

Driving to a location defined by its coordinates

If this drive is performed in isolation, i.e., from a stationary position to a stationary position (e.g., drive from the digging position to the processing plant), the rover first turns in place to reach teh the yaw pointing towards the destination, then it drives straight (while avoiding obstacles) to the desired coordinates, then it turns in place to the yaw desired at the destination.
If this drive to a specific location is a segment of a larger travel plan (e.g., scout's exploratory path), we skip turning in place and instead drive forward while steering left or right (within the maximum angular velocity parameter) as needed to point towards the destination.
As we do not build a global map of the work area, we do not perform any path planning.

Visual approach

While we are maintaining the location of each robot, during the 2 hour run inaccuracies of up to 4-5m of difference from a true rover's pose can develop. Therefore, when a precise alignment of two rovers or a rover and a lander is needed, we direct the rover to drive to a nearby location based on the target's coordinates, then the rover finalizes the approach based on visual feedback from its cameras, using YOLOv3 detection and/or additional computer vision algorithms (edge detection, etc).

Terrain specific driving and emergency recovery

Under normal circumstances, all rovers move at their maximum velocities. As they travel they monitor their own pitch and roll. If either of these exceeds a preset threshold, the rover slows down to about half its maximum velocity. This helps to navigate a sloped terrain more safely given that abrupt changes of trajectory while traveling on sloped terrain could lead to the rover overturning. When traveling at lower speeds, any urgent changes in driving trajectory do not result in as abrupt a change.

If despite all precautions the roll or pitch of a rover exceeds preset thresholds representing a critically excess tilt, the rover performs an emergency downhill move until both roll and pitch get to within their safe limits. This is done by calculating the direction pointing downhill and then driving in that direction at maximum velocity in the emergency mode (see Swerve driving).

Excavator arm controller

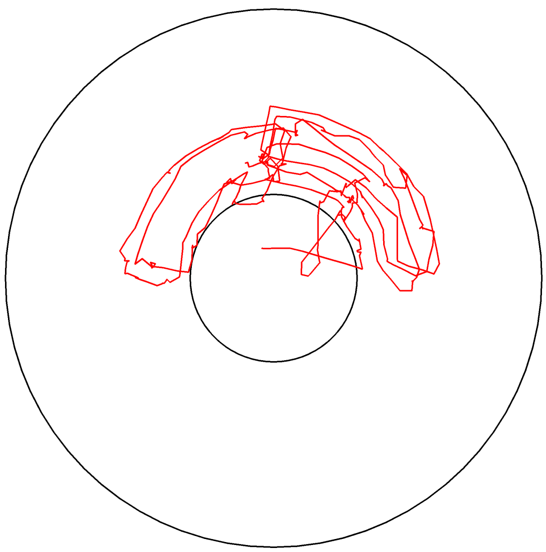
Our system defines desired movements of the excavator arm by providing angles for all four joints to be reached. We use a module from GummiArm project[7] to convert these directives into real time commands for the individual joints. We use the excavator's pitch and roll to adjust the movements such that the bucket stays horizontal while transferring the volatiles from the regolith to the hauler's bin and also to adjust the position of the bucket when the volatiles are released so that the volatiles fall inside the hauler's bin.

Scout exploration

Each team (and therefore its scout) is responsible for exploration of half the workspace, i.e. a semicircle formed by dividing the workspace in half along the x=0 axis. This area is divided into four arcbands, each π/2 wide and inner-outer radius of either 30-60m or 60-90m. Hence, four such arcbands cover the assigned half of the workspace where volatiles are found, i.e. an arcband of π length with inner radius of 30m and outer radius of 90m. Given our overall strategy, most driving and digging occurs within the two arcband segments closer to the center.

[7] https://github.com/GummiArmCE/gummi_interface/blob/master/scripts/follow_joint_trajectory.py

Each segment is explored by the scout by driving on concentric arcs from one end of the arcband to the other, with 4m distances between adjacent arcs. Given 2m reach of the volatile sensor, targeting 4m between parallel paths is our choice for balancing the need for not missing any volatiles along the path with having too much overlap between neighboring paths and hence not exploring the overall workspace fast enough. Of course, the actual path of the scout will differ from these geometric shapes on account of obstacles on the path that need to be avoided as well as because of inaccuracies in the maintained current position.

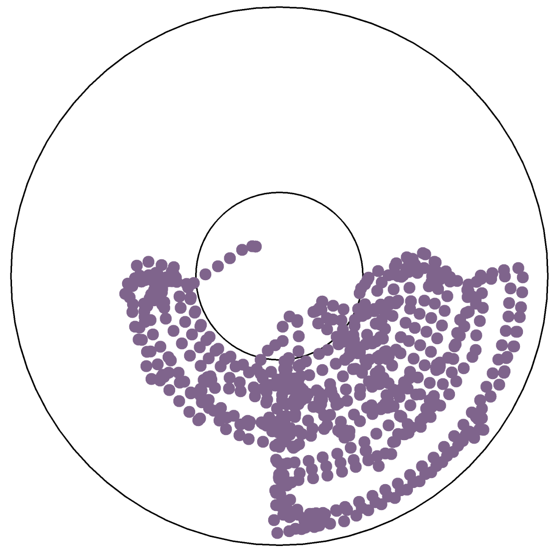


Team 1 exploration pattern

Team 2 exploration pattern

For example, after about 1h30 of simulation, we can see that Team 1 (exploring the upper semicircle) is still working within the two arcbands closer to the center while Team 2 is already working in one of the outer arcbands. This is because in this particular seed, there weren't as many volatiles found closer than 60m to the center in the lower semicircle as there were in the upper semicircle. Hence the scout in the upper semicircle spent more time assisting with the digging and didn't proceed to the farther arcbands by this time. Also note the paths leading from the segments towards the center of the circle, these indicate scout's interruption of the exploration pattern by a trip towards the recharging station.

Team 2 coverage pattern



Team 2 “thin” coverage pattern

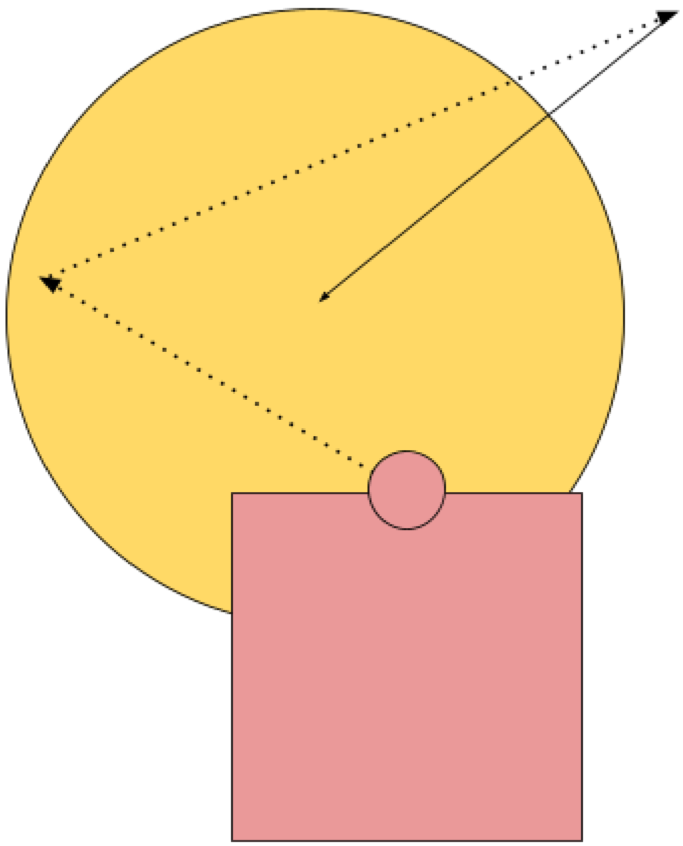
Further, we can verify how well we are exploring the given area, i.e. how much space we are leaving unexplored vs. how much space we are covering more than once. To observe this, we look at the coverage and “thin” coverage charts. In the coverage chart, any white space left behind indicates an unexplored area. The “thin” coverage shows what would happen if the scout’s sensor had a slightly shorter reach than 2m. In this diagram we expect slight gaps between parallel arcs. If there is no gap, it means that we covered the same area more than once. While we see slivers of while space in the coverage diagram and some areas covered twice on the “thin” coverage diagram, we feel that our solution is striking a good balance given the constraints (location accuracy and obstacle avoidance needs).

Volatile center discovery and volatile acceptance

While the scout drives around it monitors its volatile detector readings. As soon as a volatile is detected, it performs a quick zig zag pattern (0.8m diagonally in one direction, then 1.6m diagonally further in the opposite direction) while capturing its location and the distance to the volatile in each sensor cycle. Typically, such an exercise yields a few dozen data points. We then use trilateration[8] to calculate the center of the volatile, then we calculate its offset from the

[8] https://www.alanzucconi.com/2017/03/13/positioning-and-trilateration/

scout's current position and drive the scout to it. In an ideal case, the distance of the sensor to the volatile at the end of the offset move should be 0m. We then move forward by the distance between the sensor and the center of the robot thus positioning the center of the robot over the center of the volatile.

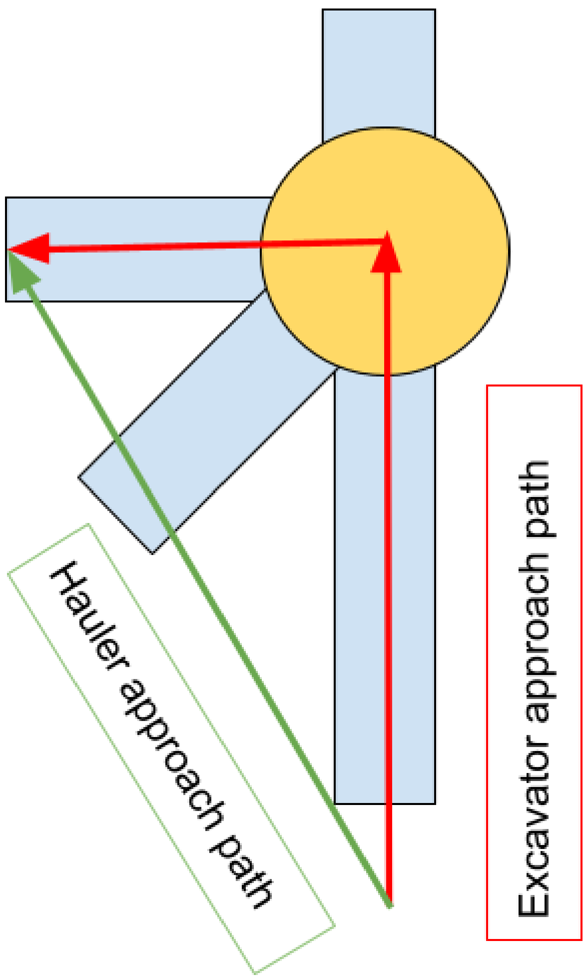


The yellow circle represents the volatile. The red square is a robot with its volatile sensor indicated by the red circle. Once the sensor detects a volatile, it performs diagonal movement to the left followed by diagonal movement to the right while capturing volatile distances in each sensor detection cycle. It then performs trilateration on the collected data to calculate the coordinates of the center of the volatile and attempts to move the robot such that the sensor is right over the volatile center (i.e., the measured distance should be virtually zero meters at that point).

If any of the attempted moves are not possible due to obstacles in the area or if during this operation either pitch or roll exceed a given threshold, this volatile is deemed **inaccessible** and the scout continues to look for the next one. Otherwise, the scout waits in place for the arrival of the excavator (while charging its batteries).

Scout approach pattern selection

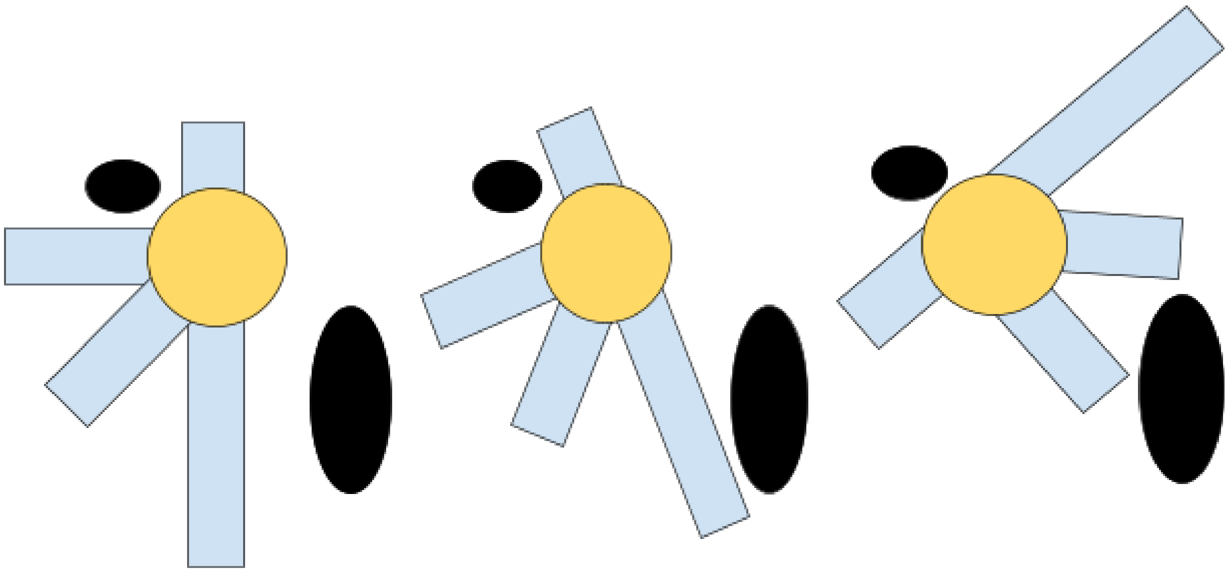
Having built and maintained the local map of obstacles, at the end of its volatile discovery the scout identifies all the possible approach directions for the excavator and hauler (i.e. those approach patterns without obstacles near the scout/volatile). This list of approach directions is then passed to the overall solution controller which uses it to decide on the best direction the excavator and hauler should approach this volatile from.



The yellow circle indicates a volatile. Upon centering over a volatile, the scout finds all yaws where the pattern indicated in blue is obstacle-free based on its local map.

The longer bar pointing down represents the path along which the excavator would approach the volatile. The bar pointing to the left is a path that the excavator will take to move slightly off the volatile in order for its bucket to dig from the center of the volatile. The diagonal bar is an approximation of an area that needs to be free for the hauler to be able to approach the excavator. The hauler will be approaching from the same direction as the excavator but by then the excavator will be slightly to the left of the volatile.

Finally, the bar pointing up indicates a path that the scout will initially take to move on from the volatile.



Black ovals indicate obstacles detected by the scout. The scout identifies these three potential approach patterns for excavator and hauler that would avoid the obstacles. Later, when the excavator is ready to approach this volatile, it is up to the central controller to decide which of these safe approach patterns to use based on the location of the excavator. The orientation that results in the shortest travel distance for the excavator is used.

Scout active wait

While the scout is waiting for the excavator to arrive, if it is standing even on a modest incline, it can slowly slide downhill from the optimal position over the volatile during the wait. To offset this, we are monitoring the distance to the volatile during the wait and if it increases by more than a threshold parameter from the initial distance, we let the scout drive uphill for that distance hence getting approximately where it started the wait.[9]

Scout-excavator volatile hand-off

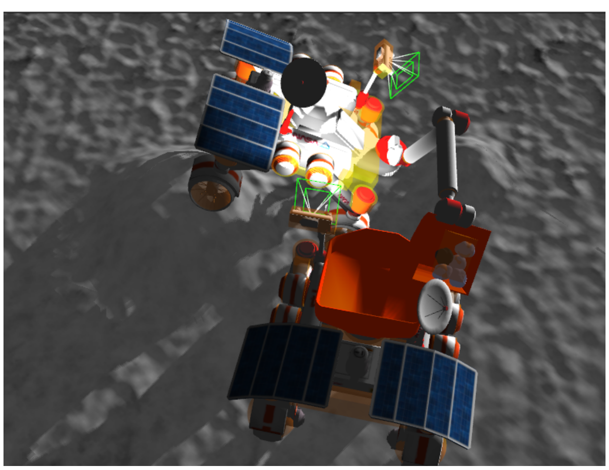
The goal is for the excavator to replace the scout on top of the volatile, then move sideways a little so that it is actually the bucket, not the body of the excavator that is closest to the center of the volatile. To that end, the excavator is directed to drive to a location about 8m away from the scout using the coordinate system. Once it arrives, it is directed to drive visually towards the

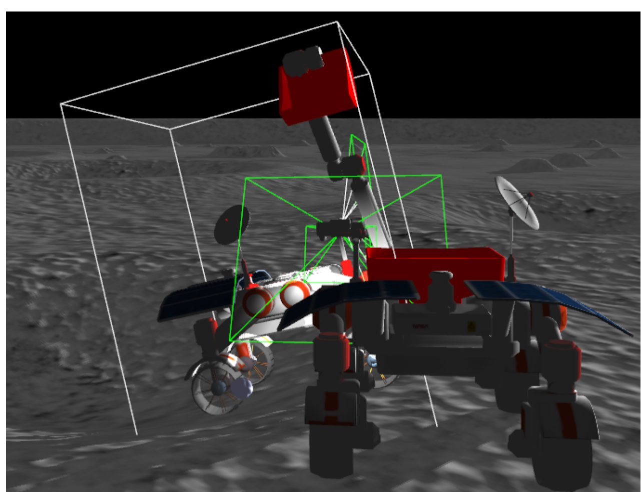
[9] We were getting better results measuring the change in volatile distance over measuring the change in the internally maintained rover position.

scout to a distance of about 3m. The scout should be visible in the beginning of the visual approach but some sideways moves may be needed to align the excavator right behind the scout while maintaining the desired yaw. Once the excavator is 3m away from the scout, the scout is instructed to continue searching for the next volatile (and to reject any volatile closer than 12m from the current position to avoid this scout interfering with the upcoming volatile extraction). Once the scout is at least 8m away from the volatile, the excavator drives exactly 3m minus the length of a rover forward (which should put it right over the volatile where the scout was waiting), then performs a slight offset move to center its bucket over the volatile instead (see Scout approach pattern selection).

Excavator-hauler approach

Once the next volatile is found by the scout, the hauler immediately drives towards it until it is about 30m away, it stays put if it is within 30m already. This is in order to preposition the hauler near the volatile but in the same time not to have a second rover in the area where the excavator will be visually approaching the scout. Once the excavator is in place, the hauler drives to a location about 8m away using the coordinates, then it switches to the visual mode and drives towards the excavator until it is about 3m away (avoiding obstacles in the process). Once it reaches that destination, it proceeds to drive towards the excavator (ignoring any perceived obstacles because this area should have been pre-cleared by the scout) until it reaches the target distance of 0.8m which is the distance for which the volatile transfer was designed for. The hauler needs to be aligned with the excavator such that the excavator joints are directly in front of hauler's left camera. This alignment results in the best position of the hauler's bin vs. the excavator arm's shoulder yaw joint and one that the volatile transfer was designed for.

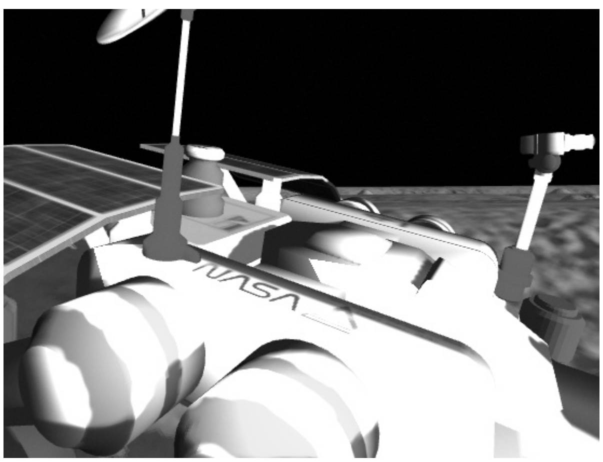




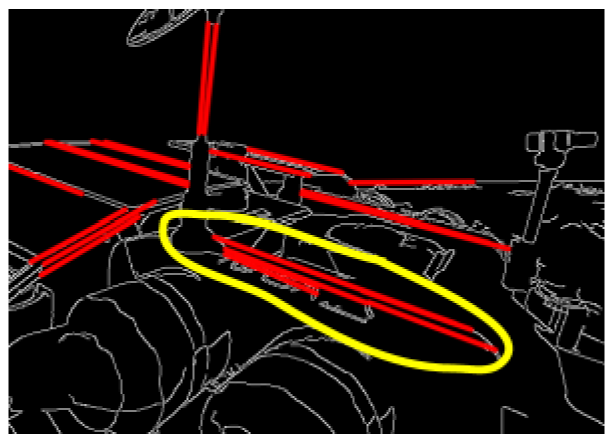
The hauler is positioned perpendicular to the excavator offsetting the pitch of the excavator, due to either standing on a boulder with its rear wheels (left) or standing on the slope tilted backwards (right). The excavator is taking into consideration its:

1. Pitch - to adjust the shoulder yaw joint angle (if the pitch points down, the shoulder yaw joint needs to point further backwards)
2. Roll - when adjusting how far from the excavator will the arm stretch out (if roll to the right, we drop the clods closer, if roll to the left, we stretch the arm out farther)

Finally, the hauler then looks at the top edge of the excavator's body adjusting for excavator's pitch and its own roll and if the edge does not appear to be horizontal, the hauler turns slightly until the top edge is close to looking horizontal on the hauler's camera. This assures the body of the hauler is positioned perpendicularly to the body of the excavator which is needed for optimal volatile transfer performance.



Original view of the joints



Same view with straight lines detected. The longest one above joints (highlighted in yellow) is used to determine relative orientation of the excavator and the hauler

Hauler maintaining distance

Given that the (modeled) lunar surface is somewhat slippery, robots tend to slowly slide when standing in place for a while. To avoid having the excavator and hauler slide out of their alignment during the loading process, the hauler adopts an active wait where it maintains a fixed distance from the excavator, going slowly backward or forward as needed. This is achieved by having its sensor lock onto the joints of the excavator, i.e., by continuously modifying the sensor bar pitch such that the horizontal centerline of the joints stays in the center of the camera view. The minimal distance measured by the lidar within the bounding box of the detected joints object is then acted upon in terms of driving forward or backward to keep it constant.

This maintaining of distance between the two rovers prevents them from sliding into each other or, on the other hand, from sliding away and failing the volatile transfer. This does not avoid sideways sliding (from the perspective of the hauler) but often excavator and hauler would slide in the same direction hence still maintaining their relative position, even though extreme cases of sideway sliding could result in moving outside of the volatile. In such a situation, not all volatile may be retrieved from this location.

Visual approach recovery

Despite numerous precautions to minimize failed visual approaches, they can still occur. For example, when an obstacle appears 3-4m away from the target rover, it may throw the approaching rover off its path and the approach is not successful, the approaching rover avoids both the obstacle and the target rover and instead drives past it.
We detect this scenario by noticing either 1) the visual detection of the rover disappeared or 2) a rover was suddenly detected much further away (this can occur if the nearby rover is no longer visible but there is another rover in the distance).
In this situation, the approaching rover turns around 360 degrees and maps all rovers around. It then identifies the closest one as its most likely target and performs a circular move around it (similar to the processing plant hopper approach) until it is within the desired distance and with the desired yaw. This area close to the target rover should be relatively clear of obstacles due to scout's pre-screening of the volatile area so it is a relatively safe operation to perform.

Digging sequence

Once the hauler successfully approaches the excavator (its left camera is looking directly at the right side joints of the excavator) and the relative orientation is as close to 90 degrees as possible (see Excavator-Hauler Approach), the Hauler Maintaining Distance process is started. At this point the excavator starts digging exactly in front of itself (arm shoulder yaw is zero) and then, if any volatile is inside its bucket, drops the load at 90 degree angle to the right (adjusted for the excavator's pitch). While the downward move during the scooping operation is always the same in terms of individual arm joint angles, the arm joint angles during the volatile drop may differ somewhat based on the roll of the excavator. If it rolls to the right, the arm will extend less, if to the left, the arm will extend more in order to release the bucket over the hauler's bin. After the bucket is opened and its content dropped, the excavator proceeds to perform another dig.

If there are no volatile clods inside the bucket once it is retrieved above surface, this volatile location is deemed exhausted, the bucket is emptied outside the hauler bin and the digging process stops. In some cases the volatile may not be actually exhausted, we have experienced scenarios where no volatile clods are retrieved even though the volatile location is exhausted. Same result can also happen if the excavator slides off of the volatile location during the excavation and loading process.

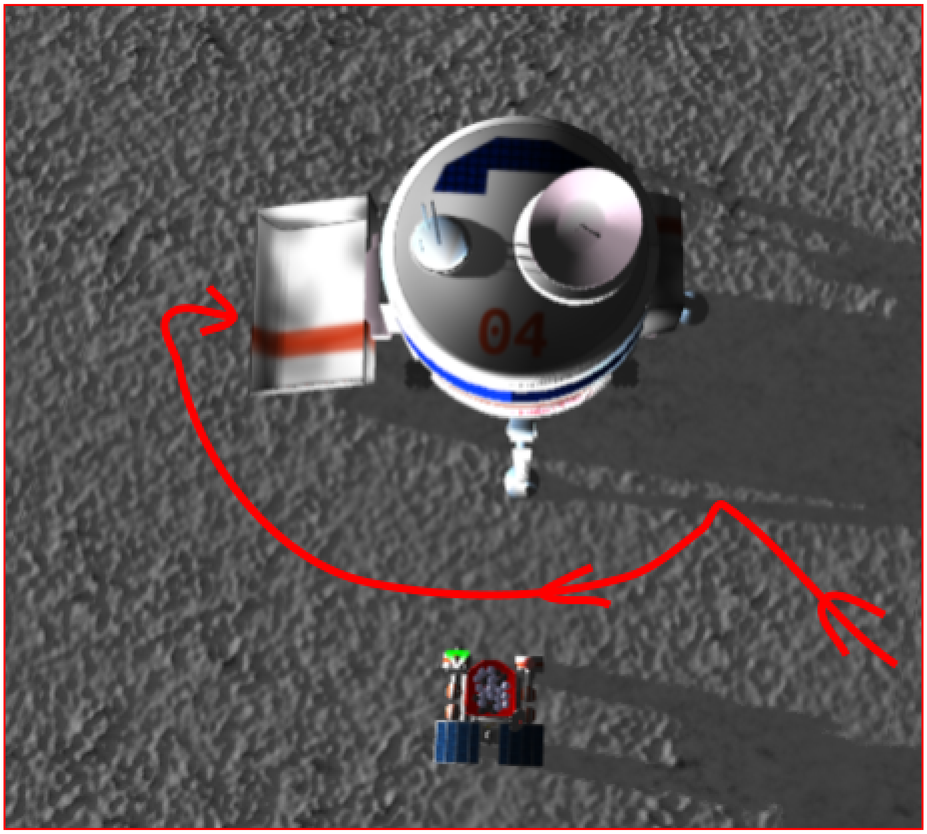
Hauler hopper approach and unloading

Once a volatile is exhausted as detected by the excavator (all clods in the latest scoop are regolith), the hauler pulls back from the excavator by 2m and then it proceeds towards the processing plant (assuming at least two scoops were dropped into its bin, otherwise it stays put and is ready to handle the next volatile). The first part of the journey is driven based on the coordinates (to avoid relying on visual navigation from distant areas of the map where the lander detection may be inaccurate and/or intermittent). Once the hauler arrives about 30m from the processing plant[10], it switches to visual navigation. Once it arrives about 10m away, it should be in an area without any obstacles. It then changes the tilt of the sensor bar to point about 30deg upwards so that its laser is hitting the cylindrical portion of the processing plant (i.e., not the hopper, not the legs). It then continues to approach the processing plant until the laser indicates the nearest distance of 3.2m.

At this point the rover will circle around the processing plant while facing the plant until the hopper is right in front of it. The direction of the circular drive is determined by the current yaw of the hauler, if it is between 0 and π, it will move left, if it is between 0 and -π, it will move right. The rover controls all degrees of freedom to maintain:

1) Distance from the processing plant
2) Center of the processing plant in the center of the view[11]

[10] These distances are estimated based on the YOLO bounding box of the detected objects.
[11] We estimate the center of the cylinder from the laser detections which should form a semi-circle.

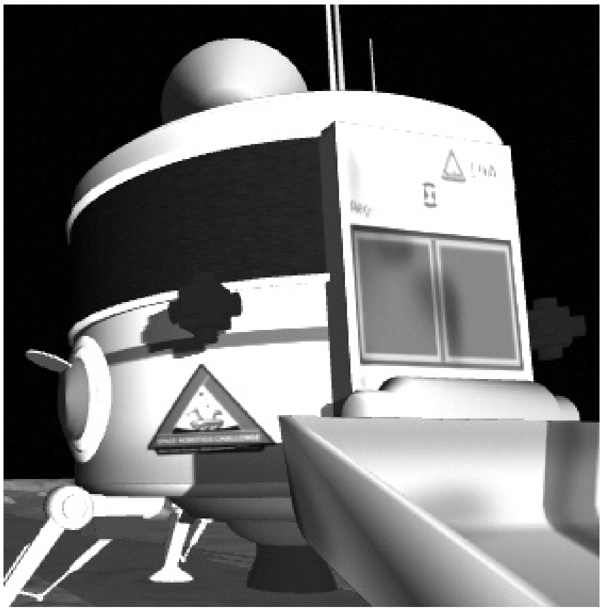


Hauler is circling around the processing plant while maintaining distance from and orientation towards the center of the processing plant until alignment with the hopper is detected.

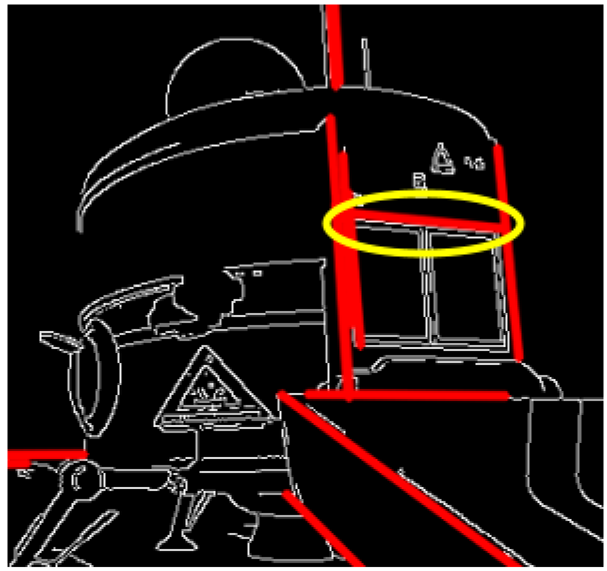
As the hauler drives around, once it detects the gray rectangle above the hopper within its view, it will start evaluating its position and orientation relative to this rectangle (and therefore the hopper) using the following methods:

1) Is the camera pointing towards the center of the processing plant cylinder?
2) Is the gray rectangle above the hopper in the center of the view?
3) The gray rectangle above the hopper laid flat has ratio of its sides 1:1.4. What is the ratio of the sides of its YOLO bounding box now?
4) If the ratio from the previous item is close to 1:1.4, calculate the slope of the upper edge of the gray rectangle. Slope value of zero (horizontal line) indicates that the rectangle is right in front of the camera, non-zero numbers indicate the rectangle is observed at an angle.

We determine that the hauler is sufficiently aligned for unloading if the four above conditions are met. In that case, the circling motion stops and the hauler proceeds to go straight until it reaches the exact distance of 2m based on the nearest laser measurement. This is the distance from the cylinder given the sensor bar pitch where the hauler bin should be emptied.



Original view of the processing station



Processing station with straight lines detected. The longest one in the vicinity of the top edge of the gray rectangle above the hopper (highlighted by the yellow oval) is used to determine alignment.

After the bin is emptied, the hauler pulls back, turns towards the repair station and drives towards it to recharge before continuing to the next volatile.

Maintaining poses

As we know, the initial locations of individual rovers as well as landers are fixed. Only their yaws change with seeds and then the yaw for all rovers is identical. Hence, we initially use one *get_true_pose* call of one of the haulers to establish a yaw of all the rovers (again, the initial locations are given).

Then, during the run, whenever a volatile is located, this opportunity is used for all the rovers of that team to sync up their locations. Specifically, once the scout positions itself over the center of the volatile, it notes its coordinates as it is tracking them internally. Once the excavator and hauler arrive at this volatile location, their poses are updated using the volatile location as determined by the scout. While this does not assure pose accuracy with respect to the global coordinate system, it assures that all robots have accurate relative positions. Hence, we can successfully plan for one robot to arrive at a given location with respect to another robot (e.g., an excavator will accurately arrive 8m behind a scout).

Based on our observations, the IMU functionality is sufficient to maintain accurate enough yaw during the whole run so once each rover's yaw is initially established, it is not changed outside the normal IMU-based calculations.

Although globally accurate poses are not critical for robot collaboration or interaction with the landers, there is still some value in keeping the inaccuracy with respect to the global coordinate system within reasonable bounds. One reason is to make sure that scouts' exploration patterns are indeed covering the work area as intended. It also helps the system to work more efficiently as the visual operations are better prepared with more accurate coordinates-based approaches (e.g., during the drive towards the recharging station which starts with a coordinates-based stretch followed by a visual approach once the rover is closer by). To that end, after one simulated hour (half way through the run), each scout will request its own *get_true_pose* next time it stands over a volatile, then the other robots update their poses from these scouts to be globally accurate as well.

Team interaction

There is no collaboration between the two teams, each operates independently of the other. A 6m buffer separates both semi-circles of their respective work spaces to minimize interference between both teams. The only scenario where there is an interaction is when a hauler is delivering volatiles into the processing plant hopper. We utilize a simple semaphore where before a hauler attempts to perform the final approach towards the processing plant, it checks the semaphore. If it's unlocked, it locks it and proceeds to approach the processing plant, circle around until the hopper is found, drops the volatiles and moves outside the immediate area around the processing plant, then it unlocks the semaphore again. If the semaphore was locked (i.e, the other hauler is already in the process of delivering), this hauler waits until the semaphore opens up.

III. RESULTS AND DISCUSSION

We feel we have achieved a sound solution which seems to be robust and efficient. Rovers move at their respective maximum velocities, terrain permitting. The operational strategy seems efficient albeit not mathematically proven as optimal. The solution appears to regularly operate at full capacity for the whole two sim-hours of the run albeit given that a 2-sim hour test takes more than 24 hours to execute, we were only able to run such a complete test of our final version several times.

If the volatile distribution is unfortunate enough that the scout - following its prescribed exploratory pattern - doesn't detect any volatile for a while or all volatiles it detects are in inaccessible areas, its teammates may be waiting unnecessarily, however we find this scenario is not that common. More typically volatiles are detected regularly enough that the excavator and hauler are utilized close to capacity.

Our method for positioning the excavator over a volatile seems to result in a highly efficient digging process as we dig up the available maximum of five volatile clods in most scoops hence often needing only four scooping motions and certainly no more than five[12]. The additional benefit of this efficiency is that the hauler is not particularly full when delivering the volatiles to the processing plant hence driving over uneven terrain doesn't typically result in any spilling.

It also seems that we have reached a good division of labor among the robots as often none of them needs to wait for another one too long (e.g., the excavator for its hauler to arrive, the hauler for the excavator to arrive to the volatile, the scout for the excavator to arrive to replace it over the volatile, etc).

There are indeed certain intentional inefficiencies built into the solution. For example, for any visual driving towards and approach of a rover, we aim to have only that rover anywhere near the area so that the arriving rover cannot get confused about which of the multiple rovers it sees it is supposed to visually approach. We also try to minimize situations where two rovers operate in the same area in independent capacities (e.g., a scout is exploring while an excavator is digging in the same area). These precautions result in more driving compared to a less conservative approach.

Further optimization could probably be achieved by shrinking buffers, delays and other safety mechanisms while making sure robustness of the solution is not compromised.

While the exact scores of individual runs differ due to the different distribution of volatiles and variation in the terrain generated by different seeds which makes some of our operations easier or harder and hence more or less time consuming, our complete solution based on two teams, each consisting of one robot of each type, tends to score around 400 points in a full 2-hour run (i.e.,100 points per team per hour).

The point gain is not evenly distributed throughout the 2 hour period obviously, initially only the scouts are driving and the remaining rovers are waiting for the first volatiles to be found. Further, minimum amounts of delivered clods need to be reached for some of the volatile types before

[12] In reasonably flat terrain. On sloped terrain, the excavator may slide slightly before and during digging, decreasing the effectiveness of each scoop.

points are awarded. The solution only reaches its maximum yield per minute about 30 sim minutes into the run (depending on how long it takes to find the first volatile for each team). Then it takes about an hour for each team to clear their respective first two segments closer to the center (i.e., four segments which combined form a circle around the center of the workspace with a radius of 60m) which are the most efficient to work on given the shorter distances that need to be traveled for recharging and for volatile delivery.
Then the remaining time is spent in the farther two segments (which also have a larger area than the closer segments). In those the excavation rate is somewhat slower as the hauler needs to drive further to deliver the volatiles.
By the end of the 2 hour run, typically only one of the farther segments for each team is somewhat covered.

A sample result at the end of a 2-team, 2-hour run may look like this:

```
[rosout][INFO] 2021-06-17 15:58:43,093: dispatcher: score: 403;
hauler_volatile_score: 0.0
[rosout][INFO] 2021-06-17 15:58:43,094: * ammonia: 38.5
[rosout][INFO] 2021-06-17 15:58:43,094: * carbon_dioxide: 6.3
[rosout][INFO] 2021-06-17 15:58:43,095: * ethane: 2.7
[rosout][INFO] 2021-06-17 15:58:43,095: * hydrogen_sulfite: 42.0
[rosout][INFO] 2021-06-17 15:58:43,096: * ice: 2170.0
[rosout][INFO] 2021-06-17 15:58:43,096: * methane: 1.0
[rosout][INFO] 2021-06-17 15:58:43,097: * methanol: 1.8
[rosout][INFO] 2021-06-17 15:58:43,097: * regolith: 281.0
```

Several videos highlighting specific operations or capturing the whole 2-hour runs are available in this playlist:

NASA SRCP2 - Team Robotika Finals
https://youtube.com/playlist?list=PL3KtjefdzWVmwn2tKYIgEZvPk7yYgILaE

IV. CONCLUSIONS

In conclusion, we would like to highlight the following decisions and ideas behind our solution:

1. Robot localization - based on visual odometry, wheel odometry and IMU yaw, typical accuracy within 3m from the true pose
2. Only local maps for each rover, no global map
3. Systematic work area coverage (by scouts) - cover the whole work area using a pattern such that when a volatile is found, exploration that immediately follows avoids the area nearby (in order for this scout not to interfere with the loading setup by its teammates)

4. Volatile screening - if volatile is detected but the surrounding area is deemed to be too treacherous, the volatile is ignored.
5. Accurate approach of an object - drive to a nearby location based on coordinates, then approach visually the rest of the way
6. Safe terrain navigation - use lidar to scan the terrain, identify dangerous areas and enter them into the local map for the driving module to avoid; emergency downhill drive as a fallback
7. Safe loading approach and configuration - use scout's observation of the area around a volatile to determine whether it is safe to attempt to dig it up and if so, identify all directions from which the other rovers can approach
8. Stable and efficient loading - use roll and pitch of both excavator and hauler to determine the exact arm joint angles for best efficiency. Maintain distance between both rovers dynamically using lidar distance measurement.
9. Power management - whenever a rover is waiting, it turns to charge off of its solar panels. If battery level drops below a threshold while on the move, a detour towards the repair station is performed.

Finally, we would like to thank NASA and the whole team behind Space Robotics Challenge Phase 2 for organizing such an exciting project and we hope our participation will contribute to the future of space exploration. We would be happy to talk to the organizers further about our solution, ideas and insights we gained into solving robotics missions in space.