Time-Dependent Planning for Resource Prospecting

Christopher Cunningham[1], Heather L. Jones[1], Jacqueline Kay[2], Kevin M. Peterson[3], William L. "Red" Whittaker[1]

[1]Carnegie Mellon University, Pittsburgh, PA
e-mail: ccunningham@cmu.edu, hlj@cs.cmu.edu, red@cmu.edu

[2]Swarthmore College, Swarthmore, PA
e-mail: jacquelinekay1@gmail.com

[3]Astrobotic Technology, Inc., Pittsburgh, PA
e-mail: kevin.peterson@astrobotic.com

Abstract

Future planetary robotics will require path planning that ensures rover safety while responding nimbly to new information and evolving goals. Such missions include prospecting for ice at the lunar poles. These challenge robotic explorers in ways not encountered in equatorial missions that benefit from high solar elevations. Long-distance traverses must be achieved over short timescales. The grazing sun angles cause substantial, time-varying shadows that require paths to consider rover temperature and power balance in addition to terrainability. This paper details a hierarchical planner that considers these elements to rapidly generate long-distance paths. The planner has two components: a high-resolution planner that pre-computes feasible trajectories over short distances and a low-resolution mission planner that leverages the results of the kinematic planner to quickly find long-distance, feasible paths. Planner capabilities are demonstrated in simulated traverses on the poles of the Moon.

1 Introduction

Future robotic planetary missions will visit extreme environments that demand mission planning software to rapidly adapt to new terrain, illumination, and mission goals. Such missions include prospecting for ice on the lunar poles and circumnavigation of a planetary body. Current Mars rover missions have lasted years and the relatively stable mid-latitude environment has allowed rover operators ample time to plan operations. A mission to the lunar pole to prospect for ice may last less than two weeks due to the extreme thermal cycles on the Moon, dynamic shadows, and grazing sun angles. In the time that it took the Mars Exploration Rovers to get off their landers, a polar rover might have to complete its entire mission. In addition, the cadence of lunar operation is fast due to short communication delay and high situational awareness. This paper details a path planner that addresses the challenges faced by a robotic mission to the lunar poles to rapidly plan and re-plan paths that consider energy, temperature, and time. The planner has a hierarchical structure. It caches high-resolution, local data over short segments for use in a low-resolution, long-distance planner that can rapidly re-plan in response to changing objectives in a fast-paced mission.

The LCROSS impact confirmed the existence of polar ice, but driving and drilling are still required to determine composition and measure distribution and concentration [1] [2]. A mission to map lunar ice must be rapid and precise. The operational window for a drilling mission is approximately ten Earth days, due to lunar daylight hours spent on dawn, landing, and dusk [3]. Survey sufficient to prove an abundance of ice will require samples over multiple kilometers, requiring the ability to drive intentionally and efficiently to precise drilling destinations. Polar operations encounter time-varying, low-angle lighting; this creates shadows which are important for volatile accumulation but confront robot operations with challenges in power production and thermal control [4].

Polar resource prospecting missions must adapt mission plans in real time by reacting to data gathered during operation. Due to limited mission time and changing illumination, even small deviations in path can have dramatic impacts on later mission objectives. For example, pausing for an unplanned regional survey after detecting high concentrations of hydrogen could mean that shadowed regions change, requiring the rover to take a longer path to reach later waypoints or having to change waypoints entirely. Because path costs are a function of time, many paths must be generated during mission planning in order to choose and sequence waypoints. This demands a path planner to be capable very quickly determining feasible paths between destinations in this time-varying polar environment.

Resource and time costs for edges depend on both the world and the state of the robot. Naïvely computing these costs at high resolution every time a long-distance path is planned is computationally expensive. This is unacceptable given the responsive re-planning needed for a fast-paced mission with evolving goals and terrain understanding. In this work, pre-computation is leveraged to speed up path planning. The world is represented using a graph where nodes represent locations and edges represent transitions between those locations. A high-resolution planner estimates the costs of trajectories between connected nodes. It computes costs for time, energy, and temperature while taking into account the kinematic constraints of the robot. This is time intensive as it must occur for every edge. However, it is highly parallelizable and need only be computed once and then stored for later reuse. Map updates to account for unforeseen geometric obstacles, slopes, or communication blackouts only affect local regions so map updates do not take much time to incorporate. The result is a set of possible trajectories between every pair of connected vertices in the graph representation of the world.

The resulting low-resolution graph with pre-computed edge costs is then used to rapidly build long paths. A long-duration, A*-based planner searches for time-optimal paths that remain within energy and temperature constraints. Dynamic state generation, state dominance, and graph pruning are used to increase the speed of computation.

This paper is organized as follows. Section 2 discusses related work in constrained mission and path planning. Section 3 outlines the low-resolution planner's search algorithm and representation of the environment. Section 4 describes the high-resolution planner and the models used for estimating temperature and energy costs. Simulated traverses on the Moon are presented and discussed in Section 5. Section 6 discusses conclusions and directions for future work.

2 Related Work

Long distance path planning for a planetary rover in dynamic polar environments requires advancements from the current state of the art. The competing demands of time, energy, and temperature require time-intensive computation of long-distance paths, while mission constraints require a quick return in order to reasses mission objectives given new information.

Prior and ongoing Martian surface science missions, like MER and MSL, had baseline mission lengths on order of months with paths and mission impacts determined on that scale. Those missions visited locations with stable, predictable environments (e.g., sunlight, power, thermal, etc.) that change seasonally [5]. The Mixed-Initiative Activity Plan Generator (MAPGEN) is used as the planner-scheduler for the MER mission [4]. MAPGEN automatically expands activities into detailed tasks and enforces constraints. Human operators specify mission waypoint goals and constraints, and the resource-aware planner then generates a path that will meet those constraints. Similarly, MSL uses the Mars Science Laboratory Interface (MSLICE) and the Robot Sequencing and Visualization Program (RSVP) to schedule tasks and enforce constraints [6]. Local paths can be estimated with the Grid-based Estimation of Surface Traversability Applied to Local Terrain (GESTALT) algorithm, and Field D* has been applied to plan automous global paths on the Mars rovers [7]. These tools work reasonably well in the Mars mission environment, where a Martian day of operation is followed by a day of planning, and the level of path planning performed is short range. In contrast, polar missions must be rapid, and the environments are less stable and hospitable. A planner must be able to quickly produce global plans that consider the dynamic nature of the environment.

Wu and Ju consider integrated path and task planning for planetary rovers [8]. They consider constraints on slope, lighting, communications, thermal control, energy resources, and data storage resources, as well as operator-imposed time bounds. They simplify the thermal constraint to a restriction on heading when the Sun is above a certain elevation. This does not map well to lunar pole exploration, where Sun elevation does not vary widely and getting too cold due to operation in shadow is a concern in addition to getting too hot. The treatment of energy is also simplified to decrease battery charge when moving and increase battery charge when sitting still, provided that the rover is in sunlight. The model presented in this paper assumes that solar power collection is independent from driving (so the rover may charge while driving), and driving at different speeds uses different amounts of power. Though they claim to take task duration into account in the planner's cost function, they do not demonstrate the ability to plan geographically different paths based on local spatiotemporal variation in lighting and communications availability [8].

Tompkins conducted research on long-term rover autonomy that maximized battery power over a traverse. The autonomy package, TEMPEST (Temporal Mission Planner for the Exploration of Shadowed Terrain) was designed and tested in two different applications: sun-synchronous navigation and Life in the Atacama Desert. TEMPEST uses deterministic planning to accomplish one or more mission goals using its models of the world and the rover, its set of possible actions, and its set of constraints including positioning, time, and power. However, while Tompkins' method claims the extensibility to handle thermal constraints, this capability is not demonstrated [9].

Though the above systems have proven effective in their applications, none address the specific challenges facing a rover navigating on planetary poles. This paper presents a planner that considers energy and temperature contraints as well as the large, time-varying shadows that characterize the environment. It leverages precomputation to quickly plan and re-plan paths between mission goals.

3 Hierarchical Planner

The hierarchical path planner presented in this paper consists of a low-resolution planner that leverages precomputed results from a high-resolution planner to quickly compute feasible paths. The time-varying environment is decomposed into an efficient graph representation that reduces detail in homogeneous regions. Energy, temperature, and time costs between nodes in the graph are estimated during a precomputation step that calls a high-resolution path planner for each edge in the graph. The low-resolution planner then uses the precomputed graph to efficiently compute paths while enforcing energy, temperature, and time constraints.

3.1 Graph Representation

The polar environment is dynamic with long shadows that vary over time. To represent the dynamic nature of the environment, the world is discretized into a time series of height and shadow maps. Each height and shadow map represents the state of the world for a discrete time interval. The duration of time intervals is chosen to be small enough that there is not substantial change in the locations of shadows between adjacent time steps. However, the time interval is coarse enough so that the robot can travel between several nodes in the same time interval before reaching the time limit and needing to reference the next time step. The height maps represent the geometry of the terrain, and shadow maps determine which regions are in shadow and which regions are lit. Shadow maps are generated by ray tracing from the sun on height maps of the environment.

A graph is used to encode the world in a representation that the planner can easily interpret. Nodes of the graph correspond to a position, (x, y), in a specific time interval, τ. τ corresponds to minimum and maximum time limits during which a particular node exists. Directed edges correspond to transitions between nodes. For each pair of connected nodes, there is a set of edges the rover can take that correspond to different changes in energy, temperature, and time. This enables the rover to change its current energy or temperature in order to remain within constraints. Unfortunately, because both temperature change and constrained energy change are nonlinear, there must be a different set of edges for each set of initial conditions. These edges and their respective costs are determined by the high-resolution planner.

The naïve representation would be to take the raw shadow maps and overlay a uniformly distributed grid of nodes. However, many regions of the Moon are fairly homogeneous with only small obstacles around which a robot could navigate with relative ease. It is the large craters and shadows that pose more of a threat and must be avoided. To take advantage of large regions with few obstacles, nodes in the graph are not homogeneously distributed. To create the graph, the shadow maps are filtered in order to eliminate most small obstacles in otherwise homogeneous regions of the world. Nodes are overlaid on the shadow maps for each timestep. This work uses a quadtree decomposition to assign locations for the nodes, which recursively eliminates superfluous nodes in homogeneous regions while adding detail in heterogeneous regions. Edges are created to connect adjacent nodes in the same timestep. The set of feasible edges and their costs for each pair of connected nodes in the graph is found by calling the high-resolution planner. Finally, edges are created between adjacent time steps by connecting nodes to their nearest neighbors in the next timestep. An example graph is shown in Figure 1.

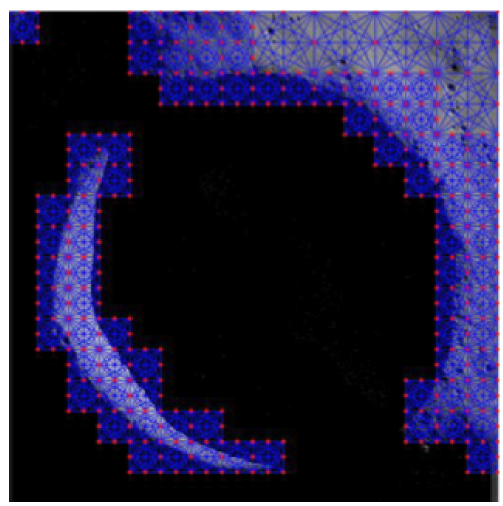

Figure 1. Graph shown for simulated imagery of Shackleton Crater on the Moon. Red dots represent nodes, blue lines are the edges between nodes. Unreachable nodes are removed.

3.2 Search Algorithm

The low-resolution planner quickly finds long-distance paths for mission planning. It uses an A*-based search algorithm that minimizes time while keeping all nodes in the path within predefined energy and temperature bounds derived from the battery capacity and operating temperature of the rover, respectively. Energy and

Algorithm 1 Path Planning Algorithm

```
openset.add(S_start)
while openset.size() > 0 do
  S_current ← openset.pop()
  closedset.add(S_current)
  if S_current = goal then
    return path
  end if
  for all edge in S_current do
    S_new ← S_current + edge
    if S_new fits constraints then
      if ∀_S [S_new > S] then
        openset.add(S_new)
        delete all states < S_new
      end if
    end if
  end for
end while
return failure
```

temperature are considered as part of search state. However, the relevant energy and temperature values are not known a priori because they are continuous variables computed during the search. Consequently, the planner does not plan directly over the precomputed nodes. Instead, it dynamically creates a new state $S = (x, y, \tau, t, T, E)$ every time it adds to the open list, where T is temperature, E is energy, (x, y) is the position, τ is the time interval, and t is a continuous variable corresponding to the time at the state. t must lie within the time constraints given by τ . A state is defined as an (x, y, τ, t, T, E) data structure that is instantiated during planning and is derived from a specific (x, y, τ) node. Multiple states can correspond to the same node. State dominance is used to prevent this method from creating an unbounded number of states. This is described in Section 3.3.

Multiple states correspond to the same node in the graph but with different energy, temperature, and time values. A state is invalid if it does not fit between the minimum and maximum temperature bounds or if it has lower than the minimum energy. When a state is created that has greater than the maximum energy limit, it is not invalid, but its energy is reduced to the maximum energy value to represent a fully charged battery.

When a state is created by following an edge, in addition to temperature and energy constraints, it must meet the time constraints of its parent-node that define the time interval in which the node exists. If does not, the state is invalid. When a state S_2 is created by following an edge e_{21} from state S_1, the time at S_2 is set to be the maximum of 1) the minimum time constraint at S_2 (t_{min}) and 2) the time at S_1 plus the time change over e_{21} (t_{nom}). Then if the time at S_2 is greater than t_{nom}, it is assumed that the rover waits and recharges for $t_{min} - t_{nom}$. Temperature and energy changes are then adjusted accordingly.

3.3 State Dominance

Dynamically creating states limits the number of states at the start of the search; however, without intelligent pruning, the number or states will grow exponentially. State dominance is used to preclude this unbounded state proliferation. New states are only created if they are strictly better than all equivalent existing states. In addition, when a new state is added, all existing states that are strictly worse than the new state are deleted from both the openset and closedset in the A* algorithm.

States are evaluated against each other based on their time, energy, temperature, and the node in the graph from which they originated. Two states are only compared if they they originated from the same node. Because energy and temperature are continuous quantities, states are considered to have equivalent energy or temperature values if the values are within δ_E and δ_T of eachother, respectively. This limits the maximum number of energy and temperature values at a given origin node to $(E_{max} - E_{min})/\delta_E + 1$ and $(T_{max} - T_{min})/\delta_T + 1$.

In order to derive state dominance, assumptions are made about the desirability of time, energy, and temperature values at a given node.

1. When all other variables are equal, it is always desirable to have a lower time cost.

2. A state S_1 is dominant over another state S_2 if S_1 has more energy and a lower or equal time cost or if S_1 has equal energy and a lower time cost.

3. States can only be compared if they have equal temperatures since there is no clear preference for higher or lower temperature within the predefined temperature constraints.

This limits the space complexity of the search by putting an upper bound on the number of active states at any given time. The theoretical upper bound is equal to the number of nodes in the original graph times a $((E_{max} - E_{min})/\delta_E + 1)((T_{max} - T_{min})/\delta_T + 1)$. In practice, however, the number of active states is generally much lower.

4 High-Resolution Kinematic Planner

The high-resolution planner estimates the temperature, energy, and time costs for all edges in the low-resolution graph. It uses physics-based power, thermal, and kinematic models in order to determine these costs. The planning algorithm is similar to that of the low-resolution planner outlined in Section 3.2. The edge costs in the high-resolution graph, however, are based on the physical models of the rover instead of precomputed values.

The graph for the high-resolution planner is discretized over a space of (x, y, θ), where θ is the heading of the rover. The state space is discretized uniformly at a much finer resolution than the low-resolution planner. Nodes are placed in a grid over the map of the environment at uniform distances for each possible value of theta. It is assumed that the resulting paths are short enough so that any change in shadow location is negligible.

Transitions between nodes in the high-resolution graph are represented by a set of edges that correspond to trajectories at several feasible velocities of the rover while traversing between the nodes. Paths between nodes and their corresponding wheel velocities are precomputed by an optimization routine using the kinematic model of the rover.

Because the temperature and energy constraints are nonlinear, costs are computed while planning in order to give accurate estimates. The models used to calculate energy and temperature changes are described in Sections 4.1 and 4.2, respectively. These calculations are one of the primary causes for the time complexity of high-resolution search.

4.1 Power Model

In this research, rovers are assumed to be solar powered with a limited battery capacity. Battery charge is decreased when the rover uses power and increased when it recharges from its solar panel. The net electrical power (p_{net}) is computed as:

$$p_{net} = p_{hotel} + p_{motors} - p_{recharge}$$

p_{hotel} is the power required to keep the rover on (e.g., cameras and electronics), p_{drive} is the power required to drive, and $p_{recharge}$ is the power generated by the solar panels. Energy expenditure in joules is calculated from p_{net} by multiplying by timestep:

$$E = p_{net}\Delta t$$

If E is positive, the rover is expending energy. If E is negative, the rover is recharging its battery. Recharge power generated by the solar panel is calculated as:

$$p_{recharge} = \phi_{solar} s \eta \cos \psi_{solar} A_{solar}$$

ϕ_{solar} is the solar flux, η is the solar panel efficiency, A_{solar} is the solar panel area, ψ_{solar} is the angle between the sun's rays and the solar panel normal, E is the solar irradiance, and $s = 0$ if the rover is shadowed from the sun and 1 otherwise.

A simplified model of rover kinematics is used to calculate driving power. The rover modeled in this paper has four wheels and is skid steered. The driving model calculates expended power based on the velocities for the wheels on each side of the rover, the slope of the ground, and other robot-specific parameters. The derivation of the model for driving power is beyond the scope of this paper.

An arc driving experiment was conducted in order to calibrate the model of driving power. A skid-steered rover was driven in a sandbox in arcs with varying turn radii. Arcs were defined by the velocity of each side, as a percentage of maximum speed. Eleven arcs were driven, varying from point turn left (right side at +50%, left side at -50%), to straight (both sides at +50%), to point turn right (right side at -50%, left side at +50%). The sum of the magnitudes of the velocities for both sides was 100% for all tests. A minimization was performed on the sum of squared error between the theoretical and experimental power to estimate the robot-specific parameters of the power model.

4.2 Thermal Model

Because there is no atmosphere on the Moon, heat transfer occurs only by conduction and radiation. This results in robot components getting very hot when they sit in direct sunlight, and very cold when they remain in shadow. Electronics have limited operating temperature ranges, so a planner must avoid overheating or excess cooling. Electronic components generate heat, so for the lunar rover considered in this paper, they are thermally coupled to a radiator plate that radiates this heat out to space. The temperature of the radiator plate is tracked using a simple thermal model, as shown in Figure 2.

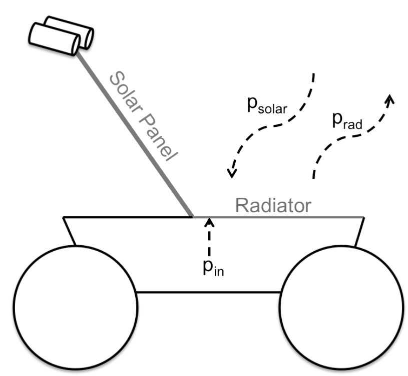

Figure 2. Diagram of heat flow

The basic thermodynamic principle used in this calculation is the definition of heat capacity (C) as ratio of heat energy transfer (ΔQ) to change in temperature (ΔT), or

$$C = \frac{\Delta Q}{\Delta T} \tag{1}$$

The heat energy transfer in a time step Δt is equal to the net heat power in multiplied by Δt. The total heat

Algorithm 2 Heat Transfer Computation

$t = 0$
$T_{cur} = T_{init}$
while $t < t_{max}$ **do**
  $p = p_{in} + p_{solar}$
  $p_{rad} = A\epsilon\sigma T_{cur}^4$
  $p_{total} = p_{rad} - p$
  $\Delta T = p_{total}\Delta t / C$
  $T_{cur} = T_{cur} - \Delta T$
  $t = t + \Delta t$
end while
$T_{change} = T_{cur} - T_{init}$

power in is equal to the heat power in from the sun, plus the heat power generated by internal components, minus the heat power radiated out to black space: $p_{total} = p_{solar} + p_{in} - p_{rad}$. The power generated by internal components is computed from the electrical power use and the efficiency of the motors: $p_{in} = p_{hotel} + (1 - n_m)\, p_{drive}$, where n_m is the motor efficiency.

Solar heat in, p_{solar}, is calculated as:

$$p_{solar} = \phi_{solar} A_{rad} \alpha \cos \psi_{rad}$$

where A_{rad} is the surface area of the radiator, α is the thermal absorptivity of the radiator, and ψ_{rad} is the angle between the sun's rays and the radiator panel normal.

The heat power into the system from internal components is computed from the electrical power used. Heat power is approximately equal to the electrical power expended for most components, except for the motors. For motors, the efficiency determines how much electrical power is converted to mechanical power and how much to heat power.

The heat output of the system is calculated by invoking the Stefan-Boltzmann law. Let ϵ be the emissivity of radiator material and σ be the Stefan-Boltzman constant. Then the heat output from the radiator is

$$p_{rad} = A\epsilon\sigma(T_{init})^4.$$

The heat transfer equation 1 is used to solve for total change in temperature:

$$\Delta T = \frac{p_{total}\Delta t}{C}$$

The algorithm used for this calculation is described in Algorithm 2.

5 Experiments and Analysis

Several sets of experiments were conducted using the hierarchical planner in order to both motivate its design and demonstrate its ability to plan paths through time on simulated traverses of the Moon.

5.1 Navigation Through Dynamic Environment

One possible future mission scenario is rover navigation around a planetary body or a feature of a planetary body such that the rover always remains in sunlight. This scenario demonstrates the ability of the planner to plan long distance traverses through considering both position and time in a dynamic world. Figure 3 shows a traverse on simulated terrain around Shackleton Crater, an impact crater at the south pole of the Moon that remains lit for a majority of the lunar year. The simulated environment was developed using digital elevation models from LRO. The path is colorized by the amount of energy. Blue and red correspond to the minimum and maximum energy levels, respectively. In addition, the path remained within the thermal constraints, tending toward the upper limit as that corresponded to driving at a higher velocity. This sample traverse shows times when the planner moves quickly because the terrain is open and others when it slows down or waits in place to charge as it waits for the shadows to move.

5.2 Running Time

One of the primary goals of the hierarchical planner design is to quickly produce feasible paths by precomputing estimates of costs for path segments using a high-resolution planner. In this experiment, a sample traverse is computed using both the hierarchical planner and only the high-resolution planner. The resulting run-times are compared. The same start and end conditions were used to the test run-times of both the hierarchical planner and the high-resolution planner. In order to directly compare results, it is assumed that the environment is static because the high-resolution planner does not support dynamic environments. Times recorded for planning only take into account the search and ignore the time it takes to construct the graph since in both cases the graph can be precomputed and cached.

In the simple environment used, both planners produced the same path, a straight line from the start to the goal. The running time of the search algorithm for the high resolution planner was 3 seconds, whereas it was only 10 milliseconds for the hierarchical planner. This confirms the assertion that the heirarchical planner saves time by caching precomputed costs for short traverses. The exact amount of time it takes to search for a path varies greatly with the environment and the constraints enforced on the search for both the low-resolution and high-resolution planner.

5.3 Energy and Temperature Constraints

The planner uses physically relevant power and thermal models in order to represent temperature and energy changes, which are then constrained to within nominal operating ranges. The simple approach would be to avoid

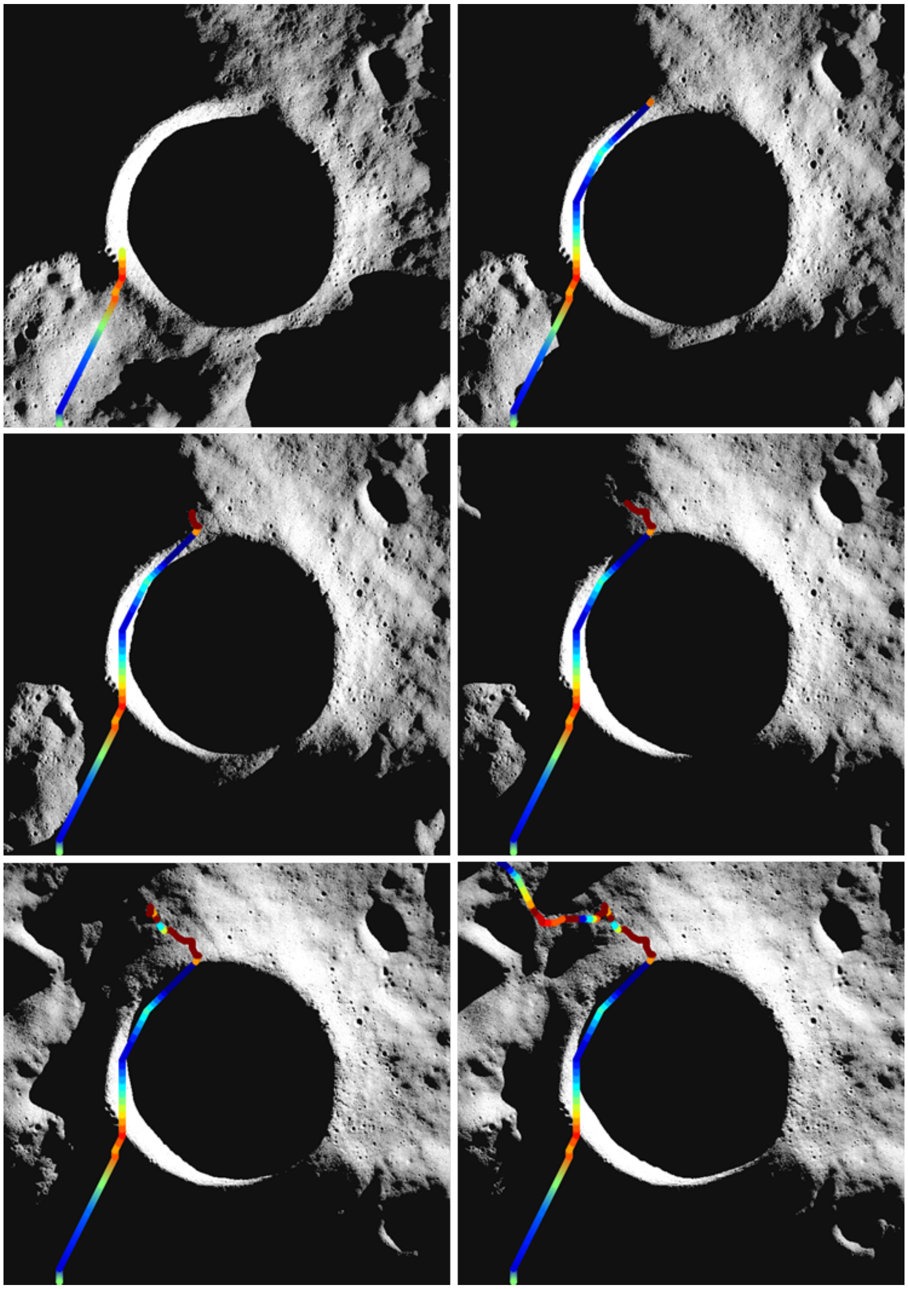

Figure 3. A simulated traverse around Shackleton Crater on the Moon demonstrating the planner's ability to plan through time in a dynamic environment. Successive images represent evenly spaced snapshots in time of the rover's path. The path is colored by the amount of energy the rover has. Blue and red correspond to the minimum and maximum energy levels, respectively.

shadowed regions entirely due to the lack of sunlight, which drains power and can subject rovers to dangerously cold temperatures. When shadows are unavoidable, then energy reserves and thermal requirements must be considered to traverse short shadowed regions.

The temperature and energy models were designed with a target rover configuration in mind. However, variables were tuned in order to produce interesting paths and see the effects of different parameters (i.e. solar panel area, radiator panel area, battery capacity, etc.) on the paths returned. In the case of both energy and temperature, it was observed that the parameters have a very large influence on paths. For example, in the limit of infinite battery capacity, the rover does not have to worry about energy and can drive through shadows indefinitely. The power model was tuned to have reasonable parameters based on an expected robot configuration. The thermal model is less straightforward, and it was difficult to tune the thermal model to produce interesting results. In most cases, the thermal configuration of the robot is either too unstable and no paths are feasible or the configuration is tuned so well that the temperature stable enough that temperature is not a limiting factor in path selection. Considering temperature and energy is critical in path planning. However, the power and thermal models must be calibrated to a particular robot for the paths to be relevant. The planner could also be useful in informing rover design through simulating possible paths.

6 Conclusions and Future Work

This paper presents an A*-based planner that leverages pre-computation by a high-resolution, local planner to quickly generate long-distance, resource-constrained traverses in time-varying environments. Caching local plans causes a drastic decrease in the time required to generate a path, enabling fast planning and re-planning during missions. Energy and temperature constraints are enforced using mathematical models of a target rover to estimate costs. These constraints are critical for finding feasible paths, and the costs are heavily influenced by robot-specific parameters in the power and thermal models. Consequently, the planner could be used to help inform rover design to ensure that desired paths are in fact feasible. Simulated traverses of the Moon show the planner's ability to plan through an environment with pitch-black, shadows that vary with time over the surface.

Currently paths are generated between one start and one goal. Further work is required to complete the full system, which will automatically select and sequence goals by running the path planner many times. How to effectively accomplish this under constraints imposed by time, energy, temperature, and rover operators is still an open area of research [10]. In addition, the planner optimizes time while constraining energy and temperature. More complex cost functions that balance preferences for energy, temperature, time, and risk should be explored in order to produce more desirable paths. Future work will explore application of this planner to longer-duration traverses including both equatorial and polar circumnavigation routes on the Moon that match the pace of Sun to enable operation through multiple lunar days without enduring the cold and dark of the lunar night. Rover missions that make seasonal forays into harsh, higher-latitude regions on Mars could also be enabled by this work.

7 Acknowledgements

This work was supported by the NASA Small Business Innovation Research program under contract NNX13CA55P. Two of the authors are also supported by NASA Space Technology Research Fellowships.

References

[1] W. C. Feldman, S. Maurice, A. B. Binder, et al. Fluxes of Fast and Epithermal Neutrons from Lunar Prospector: Evidence for Water Ice at the Lunar Poles. *Science*, 281(5382):1496–1500, September 1998.

[2] Anthony Colaprete, Peter Schultz, Jennifer Heldmann, et al. Detection of water in the LCROSS ejecta plume. *Science (New York, N.Y.)*, 330(6003):463–8, October 2010.

[3] J A George, G W Mattes, K N Rogers, et al. Resolve Mission Architecture for Lunar Resource Prospecting and Utilization. In *43rd Lunar and Planetary Science Conference*, pages 4–5, 2012.

[4] Gerald B Sanders, Landon Moore, David S Mckay, et al. Regolith & Environment Science , and Oxygen & Lunar Volatile Extraction (RESOLVE) for Robotic Lunar Polar Lander Mission. In *International Lunar Conference*, pages 1–16, Toronto, Canada, 2005.

[5] M. Golombek, J. Grant, D. Kipp, et al. Selection of the Mars Science Laboratory Landing Site. *Space Science Reviews*, 170(1-4):641–737, July 2012.

[6] Navid Dehghani. Building the Ground Data System for the Mars Science Laboratory (MSL) Project: Challenges and Opportunities. *SpaceOps 2008 Conference*, pages 1–12, May 2008.

[7] Joseph Carsten, Arturo Rankin, Dave Ferguson, and Anthony Stentz. Global Path Planning on Board the Mars Exploration Rovers. *2007 IEEE Aerospace Conference*, pages 1–11, 2007.

[8] Peng Wu and Hehua Ju. Mission-Integrated Path Planning for Planetary Rover Exploration. *Journal of Software*, 8(10):2620–2627, October 2013.

[9] Paul Tompkins. *Mission-Directed Path Planning for Planetary Rover Exploration*. PhD thesis, Carnegie Mellon University, 2005.

[10] DE Smith. Planning as an Iterative Process. In *AAAI*, 2012.