Running From The Night:
Calculating The Lunar Magellan Route in Parallel

Authors: Kevin Fang and Nikolai Stefanov

1 Progress

We have figured out how to convert our Lunar Reconnaissance Orbiter LRO files, which are in .img, .image files, into a usable data format for our program and so actually it's pretty easy for us to convert these because we can use what is called the GDAL (Geospatial Data Abstraction) library, a C++ library that we can use directly to read and write a wide range of raster and vector geospatial data formats, including the .img files that we have here, and so that's going to be relatively easy for us to get slope map and visual data in. We're thinking about using it potentially for obstacle detection, if we can't find LRO files for the pre-existing crater and rock map, then we are considering if it is allowed to use an off-the-shelf machine learning algorithm for crater and rock detection to pre-analyze all the surface imagery.

2 Readjustments

We've made several design changes to our project to allow for better parallelization. We now start by processing larger images at lower resolution to quickly determine a general path. This method helps us reduce the data we need to handle right away. We are also setting up our system to preprocess this data in parallel, which should speed up our computations. For the ASTAR algorithm, we've built a version that can be run in parallel. Right now, it works sequentially, but we plan to change this by dividing the map into sections that can be processed at the same time. This includes domain decomposition, dividing the lunar map into segments processed concurrently, and parallel frontier expansion within the A* search tree, enabling multiple processors to expand search frontiers efficiently.

We've had to adjust our schedule a bit due to Carnival week (we tabled for Lunar Robotics during that time) and other commitments, which slowed us down initially. Because of this, weâre now focusing more on the essential features (how we parallelize tasks) rather than the GUI. The 'zooming' feature that lets us focus only on relevant parts of high-resolution images will be an optional optimization, with the primary goal to have parallism with MPI across both image and A* analysis.

3 Demo

For the poster session, we plan to showcase a live demonstration of the A* algorithm, running on a laptop. The fancy GUI we likely will not have time to do, with input done through terminal and outputted via text file. These live demonstrations likely will have do be done over not extreme distances due to the limited poster presentation time. We will also have charts and graphs detailing the speedup achieved through parallel processing and data locality optimizations (no preliminary figures as of now).

4 Concerns

Our biggest challenge right now is handling and analyzing the large amount of image data. We need to figure out the best way to store these large files and access them efficiently, possibly only loading what we need when we need it. This is exacerbated if we want to use GHC or PSC machines in order to run the analysis and demos – based on past experience the disk space we are allocated (as well as comp time for PSC) is relatively limited and would not be able to handle the massive lunar surface files from the LRO. We will discuss this in our meeting with course staff.

5 Schedule

Week 4 (Mid)	**Kevin:** Finalize data segmentation strategy for parallel processing. **Nikolai:** Start integrating terrain analysis with the A* algorithm.
Week 4 (End)	**Kevin:** Test initial data loading optimizations/zip using GDAL. **Nikolai:** Domain decomposition integration into A*.
Week 5 (Mid)	**Kevin:** Work on A* algorithm with segmented data. **Nikolai:** Parallel frontier expansion in A* search tree.
Week 5 (End)	**Kevin:** Debug and refine data loading processes. **Nikolai:** Finalize terrain analysis integration (crater/rock).
Week 6 (Mid)	**Kevin:** Algorithm speeds and parallel efficiency analysis. **Nikolai:** Data caching + communication overhead improvements.
Week 6 (End)	**Both:** Complete testing. Prepare the demonstration setup, including charts, laptop, and diagrams.