Mars Exploration Rover
Elevation Mapping with Noises

James Atlas
Percy (Jiaxuan) Pan, Yue Feng

5/20/2018

Summary

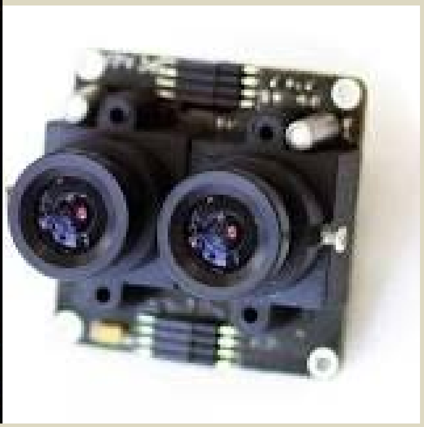
Auto-navigation is a big project for the NASA's Mars exploration rover. However, the recent Mars auto exploration robot is inefficiently and slowly running on Mars. Now, the fastest Mars Exploration Rovers are "Spirit" and "Opportunity" which have around 840 meters in 24 hours. One critical problem for the auto-navigation is the robotic environment mapping.

A mapping software which can map a precise 3D environment timely is critical for the Exploration Rover auto-navigation on Mars efficiently. Recently, the provided elevation map is limited around the robot and reflects the pose uncertainty that is aggregated through the motion of the robot. For this project, we want to design a way that can do precise elevation mapping which is taking the uncertainties of the range measurements and the state estimation. This method should be designed for navigation tasks with robots which are equipped with a pose estimation and a distance sensor (e.g. laser range sensor, stereo camera).

We choose Grid Map as our major software, Grid Map is a library with ROS (Robot Operating System) interface to manage two-dimensional grid maps with multiple data layers. It is designed for mobile robotic mapping to store data such as elevation, variance, color …. Other softwares we should use is Eigen (Linear algebra library), which can be a data type that stores grid map, user can apply Eigen algorithms directly to the map data for versatile and efficient data manipulation. Point Cloud Library (point cloud processing), which is is a ROS message type, can be used for handling the noisy from the sensor's input, and can work with the dynamic motion system.

For the algorithm, we decide the Fankhauser's research that handling the uncertainty position by using the linear prediction, their method is designed upon the grid map. The algorithm was already applied in the Grid Map library.

Because it takes a lot of time for exploration rover to scan the local environment and analyse the data, we hope our work could save some time for the robot to pick the next motion.



Specific Aims

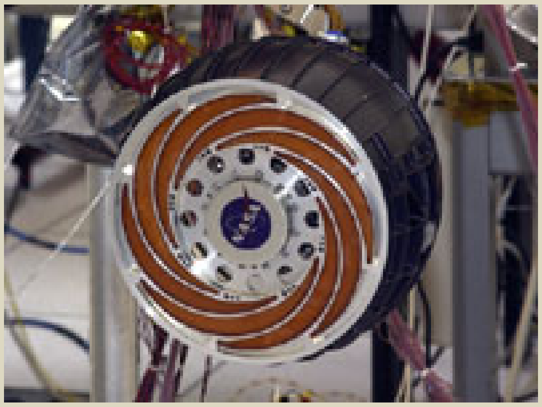
The overall goal for our research is to implement and improve the Robot-Centric Elevation Mapping for Mars exploration Rovers, which is one part of its auto-navigation system. In order to achieve the goal, our research will be realized in several phases step by step.

1. Get familiar with ROS(Robotic Operating System). Analyze the existing overview application architecture for elevation mapping, and design an appropriate application architecture for our research.
2. Compare the sensor inputs of Robot-Centric elevation mapping, and choose one or two types of sensors that we going to use in our experiments.
3. Learn how grid map library for mobile robots works with ROS. Try to input existing images and data by different sensors and output some simple 3D terrain maps.
4. Analyze Eigen which is the data type that stores information of grid map. Learn the existing algorithms, and try to improve the efficiency of the system by modifying the algorithms.
5. Analyze Point Cloud Library, which is the output type of grid map. Learn existing algorithms, and try to modify them.
6. Conclude all the experiment, and study notes. Propose some possible methods that could realize better auto driving in surface of Mars.

Background and Significance

Mars exploration Rover

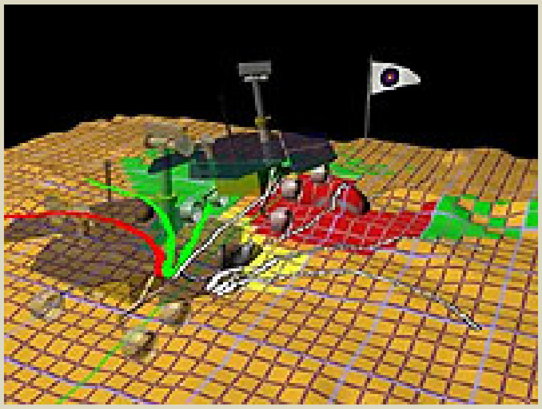
Technologies for autonomous planetary mobility enable the rovers to make decisions and avoid hazards on their own. New path-planning software has helped the rovers avoid mission barriers. When rocks are unavoidable, the inherited suspension system allows for easier maneuverability. The twin rovers were designed to traverse with a fair amount of ease over the rocky martian terrain. Sojourner's "rocker-bogie" mobility system was modified for the Mars Exploration Rover Mission.



To account for the extreme difference in weight and center of gravity from Sojourner, the mobility system on the Mars Exploration Rovers is in the back of the vehicle. The wheels are, naturally, larger and have evolved in design. Each wheel is approximately 26 centimeters (about 10 inches) in diameter and has a unique spiral flecture pattern that connects the external part of the wheel with the spoke to absorb shock and prevent it from transferring to other parts of the rover. The rocker-bogie design allows the rover to go over obstacles (such as rocks) or through holes that are more than a wheel diameter in size. Each wheel also has cleats, providing grip for climbing in soft sand and scrambling over rocks.

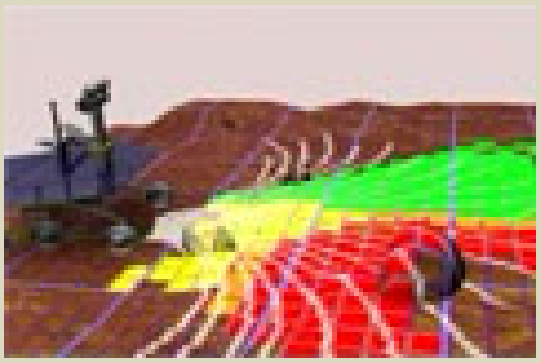
The "orange filling" between the spaces in the spiral flecture is an open-cell foam called Solimide. It was cut into crescent shapes and bonded to the wheel. Mobility engineers decided to fill in the open geometry design of the wheels to prevent rocks and debris from interfering with drive and steering actuators. Solimide maintains its flexibility even at very low temperatures so it is ideal for conditions on Mars.

The Path of Least Resistance



Having more physical capability than 1997's Sojourner rover, Spirit and Opportunity also needed more autonomy. Engineers improved the auto-navigational driving software to give the golf cart-sized explorers more freedom.

When the rovers are navigating themselves, they get a command telling them where to end up, and then evaluate the terrain with stereo imaging to choose the best way to get there. They must avoid any obstacles they identify. This capability has enabled longer daily drives than would have been possible by simply depending on step-by-step navigation commands from Earth. As of mid-August, 2004, Opportunity has used auto-navigation to drive for 230 meters (about 755 feet, or one-third the distance between Eagle and Endurance craters), and Spirit for over 1250 meters (about 8 tenths of a mile), mostly during the nearly 3000-meter (nearly 2 miles) drive to the Columbia Hills.



The auto-navigation system takes pictures of the nearby terrain using one of the Mars Exploration Rover stereo camera pairs (body-mounted hazard-avoidance cameras on Spirit, mast-mounted navigation cameras on Opportunity). After stereo images are taken, 3-D terrain maps are generated automatically by the rover software. Traversability and safety is then determined from the height and density of rocks or steps, excessive tilts and roughness of the terrain. Dozens of possible paths are considered before the rover chooses the shortest, safest path toward the programmed geographical goal. The rover then drives between 0.5 and 2 meters (1.6 and 6.6 feet) closer to its goal, depending on how many obstacles are nearby. The whole process repeats until it either reaches its goal, or is commanded to stop.

The Mars Exploration Rover autonomous driving software is more advanced than Sojourner's in several ways. Sojourner's onboard safety system also looked for obstacles, but could only measure 20 points at each step; Spirit and Opportunity typically measure more than 16,000 points from each pair of images. The average Mars Exploration Rover obstacle-avoidance driving speed of nearly 34 meters (about 112 feet) per hour is ten times faster than Sojourner's. During its entire three-month mission, Sojourner drove just a little more than 100 meters (328 feet) total. Spirit and Opportunity each broke that record in a single day; Spirit drove 124 meters (407 feet) during sol 125, and Opportunity 141 meters (about 463 feet) during sol 82.

Another improvement over Sojourner is the Mars Exploration Rover Visual Odometry software system. As the rovers drive over sandy and rocky terrains, they can slip by unpredictable amounts - even backwards when driving up very steep slopes. But the Visual Odometry system helps by giving the rover a much better notion of how far it has actually traveled. It works by comparing pictures taken before and after a short drive, automatically finding dozens of features in the terrain (for example: rocks, rover tracks and sand dunes), and tracking their motion between images. Combining that with the 3-D terrain shape is more than enough information to let the rover figure how it really moved, much more precisely than simply counting how much its wheels have turned.

Robot’s motion and 3D mapping

Harris’s three-dimensional (3D) vision system DROID (Harris 1992) uses the visual motion of image corner features for 3D reconstruction. Kalman filters are used for tracking features, and from the locations of the tracked image features, DROID determines both the camera motion and the 3D positions of the features. Ego-motion determination by matching image features is generally very accurate in the short to medium term. However, in a long image sequence, longterm drifts can occur as no map is created. In the DROID system where monocular image sequences are used without odometry, the ego-motion and the perceived 3D structure can be self-consistently in error. It is an incremental algorithm and runs at near real-time.

Thrun et al. (1998) proposed a probabilistic approach using the Expectation–Maximization (EM) algorithm. The Estep estimates robot locations at various points based on the currently best available map and the M-step estimates a maximum likelihood map based on the locations computed in the E-step. The EM algorithm searches for the most likely map by simultaneously considering the locations of all past sonar scans. Being a batch algorithm, it is not incremental and cannot be run in real-time.

Thrun et al. (2000) proposed a real-time algorithm combining the strengths of EM algorithms and incremental algorithms. Their approach computes the full posterior probability over robot poses to determine the most likely pose, instead of just using the most recent laser scan as in incremental mapping. The mapping is achieved in two dimensions using a forward-looking laser, and an upward-pointed laser is used to build a 3D map of the environment. However, it does not scale to large environments as the calculation cost of the posterior probability is too expensive.

The Monte Carlo localization methodwas proposed in Dellaert et al. (1999) based on the CONDENSATION algorithm. This vision-based Bayesian filtering method uses a samplingbased density representation and can represent multi-modal probability distributions. Given a visual map of the ceiling obtained by mosaicing, it localizes the robot using a scalar brightness measurement. Jensfelt et al. (2000) proposed some modifications to this algorithm for better efficiency in large symmetric environments. CONDENSATION is not suitable for SLAM due to scaling problems and hence it is only used for localization.

In SLAM, as the robot pose is being tracked continuously, multi-modal representations are not needed. Grid-based representation is problematic for SLAM because maintaining all grid positions over an entire region is expensive and grids are difficult to match.

Using global registration and correlation techniques, Gutmann and Konolige (1999) proposed a method to reconstruct consistent global maps from laser range data reliably. Their pose estimation is achieved by scan matching of dense twodimensional (2D) data and is not applicable to sparse 3D data from vision.

Sim and Dudek (1999) proposed learning natural visual features for pose estimation. Landmark matching is achieved using principal components analysis. A tracked landmark is a set of image thumbnails detected in the learning phase, for each grid position in pose space. It does not build any map for the environment.

Since we are concerned with locally planning the robot’s motion over or around obstacles, we are setting our focus on elevation mapping techniques, which simplify the terrain as a two-dimensional surface. Early work on the generation of elevation maps for autonomous legged robots was presented by Herbert et. al. and Kweon et. al. Their approaches build on a grid map in which each cell represents the height of the terrain in order to approximate the surface of the terrain. They use matching algorithms to find the corresponding transformation between multiple scans to build a composite elevation map, but they do not address the issues of error propagation as a result of the remaining error after matching. Cremean et. al.4 developed an approach to fuse range measurements with uncertainties into a height map. When a range measurement is taken, cells that fall into the region of the measurement are updated based on previously stored data and the uncertainty of the measurement. Cells that do not receive measurement updates are left unchanged. This approach relies on absolute position measurements from GPS and is therefore unsuitable for our application. In the approach presented by Belter et. al., a local elevation map is used that surrounds the robot and is moved along with its motion. While we use a similar setup, their approach relies on a good pose tracking algorithm and does not address the issue of a drifting pose estimation. Our approach is similar to the work of Kleiner et. al., where the elevation map is deteriorated based on the motion of the robot. The uncertainty of the robot’s position and orientation is reflected in the map by linearly growing the variance of the height estimate based on the accumulated distance and angle. This approach conservatively merges an approximation of the pose uncertainty into the height variance without taking the effect of in-plane uncertainty into account.

Grid Map

Grid Map is a C++ library with ROS interface to manage two-dimensional grid maps with multiple data layers. It is designed for mobile robotic mapping to store data such as elevation, variance, color, friction coefficient, foothold quality, surface normal, traversability etc.

Features:

- Multi-layered: Developed for universal 2.5-dimensional grid mapping with support for any number of layers.
- Efficient map re-positioning: Data storage is implemented as two-dimensional circular buffer. This allows for non-destructive shifting of the map's position (e.g. to follow the robot) without copying data in memory.
- Based on Eigen: Grid map data is stored as Eigen data types. Users can apply available Eigen algorithms directly to the map data for versatile and efficient data manipulation.
- Convenience functions: Several helper methods allow for convenient and memory safe cell data access. For example, iterator functions for rectangular, circular, polygonal regions and lines are implemented.
- ROS interface: Grid maps can be directly converted to and from ROS message types such as PointCloud2, OccupancyGrid, GridCells, and our custom GridMap message. Conversion packages provide compatibility with costmap_2d, PCL, and OctoMap data types.
- OpenCV interface: Grid maps can be seamlessly converted from and to OpenCV image types to make use of the tools provided by OpenCV.
- Visualizations: The grid_map_rviz_plugin renders grid maps as 3d surface plots (height maps) in RViz. Additionally, the grid_map_visualization package helps to visualize grid maps as point clouds, occupancy grids, grid cells etc.
- Filters: The grid_map_filters provides are range of filters to process grid maps as a sequence of filters. Parsing of mathematical expressions allows to flexibly setup powerful computations such as thresholding, normal vectors, smoothening, variance, inpainting, and matrix kernel convolutions.

Point Cloud Library

The Point Cloud Library (PCL) is a standalone, large scale, open project for 2D/3D image and point cloud processing.

Eigen Library

It supports all matrix sizes, from small fixed-size matrices to arbitrarily large dense matrices, and even sparse matrices. It supports all standard numeric types, including std::complex, integers, and is easily extensible to custom numeric types.
It supports various matrix decompositions and geometry features. Its ecosystem of unsupported modules provides many specialized features such as non-linear optimization, matrix functions, a polynomial solver, FFT, and much more.

Expression templates allow to intelligently remove temporaries and enable lazy evaluation, when that is appropriate. Explicit vectorization is performed for SSE 2/3/4, AVX, FMA, AVX512, ARM NEON (32-bit and 64-bit), PowerPC AltiVec/VSX (32-bit and 64-bit) instruction sets, and now S390x SIMD (ZVector) with graceful fallback to non-vectorized code. Fixed-size matrices are fully optimized: dynamic memory allocation is avoided, and the loops are unrolled when that makes sense.
For large matrices, special attention is paid to cache-friendliness.

Algorithms are carefully selected for reliability. Reliability trade-offs are clearly documented and extremely safe decompositions are available. Eigen is thoroughly tested through its own test suite (over 500 executables), the standard BLAS test suite, and parts of the LAPACK test suite.

The API is extremely clean and expressive while feeling natural to C++ programmers, thanks to expression templates. Implementing an algorithm on top of Eigen feels like just copying pseudocode. Eigen has good compiler support as we run our test suite against many compilers to guarantee reliability and work around any compiler bugs. Eigen also is standard C++98 and maintains very reasonable compilation times.

Preliminary Data

Kleiner et. at. developed a method for robot to deal with the autonomously in real-time which is the problem of simultaneous localization and mapping (SLAM), consisting of a continuous state estimation problem and a discrete data association problem. We will do the similar work as Kleiner's, where the elevation map is deteriorated based on the motion of the robot. The uncertainty of the robot's position and orientation is reflected in the map by linearly growing the variance of the height estimate based on the accumulated distance and angle.This approach conservatively merges an approximation of the pose uncertainty into the height variance without taking the effect of in-plane uncertainty into account.

Kalman filter tracking is a powerful tool in the computer vision, it estimates the state of a dynamic system, even if the precise form of the system is unknown. The filter can support estimations of past and even future states. The description of the standard Kalman filter and its algorithms with the two main steps, the prediction step and the correction step. Furthermore the extended Kalman filter is discussed, which represents the conversion of the Kalman filter to nonlinear systems. There are many researchers did use Kalman filter tracking to track the robotic motions in a dynamic environment, resulting in a database map with landmark positional uncertainty. Thus, there are many data from the website.

Occupancy grid mapping is an open-source algorithm library. it refers to a family of computer algorithms in probabilistic robotics for mobile robots which address the problem of generating maps from noisy and uncertain sensor measurement data, with the assumption that the robot pose is known. The basic idea of the occupancy grid is to represent a map of the environment as an evenly spaced field of binary random variables each representing the presence of an obstacle at that location in the environment. Occupancy grid algorithms compute approximate posterior estimates for these random variables. Murray et. al. did a mobile robot navigation by using real-time stereo vision, they made their robot explores its environment while building occupancy grid maps of the environment. Murray presented a method for reducing stereo vision disparity images to two-dimensional map information. Otherwise, they tried to reduce errors by segmenting disparity images based on continuous disparity surface to reject "spikes" caused by stereo mismatches. Like as following figures. After creating the map, they trained the robot to find all possibilities of ways to from the current position to the goal point. Thus, their robot got the cost map which showing the distance to the destination, and find the best way for the following moving.

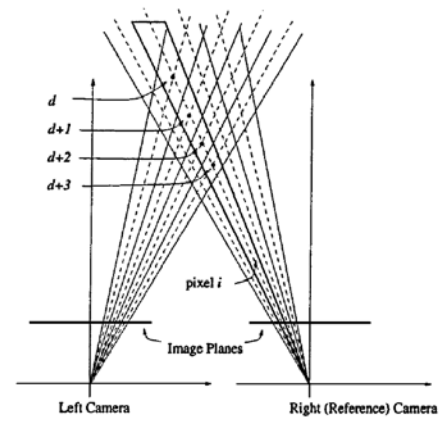


Figure 3. Stereo triangulation: dashed lines indicate lines-of-sight from the centre of the pixels, solid lines indicate lines of sight at the edges of the pixels.

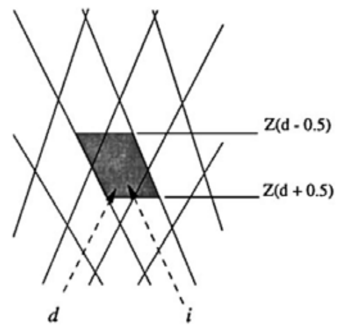


Figure 4. Region of uncertainty for a given pixel i and disparity d.

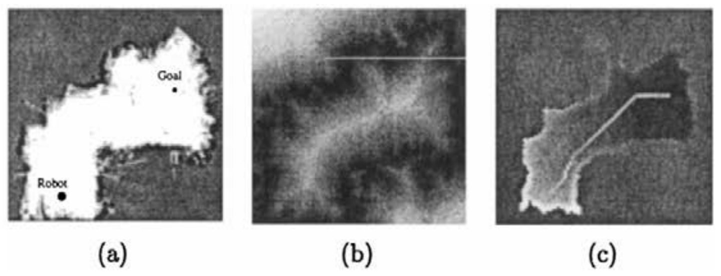


Figure 8. Path plan generation: (a) the map showing the robot position and destination (b) the distance-to-nearest obstacle map (c) the cost map showing the distance to the destination, black indicates near the goal, white indicates far from the goal, as well as the planned path in white.

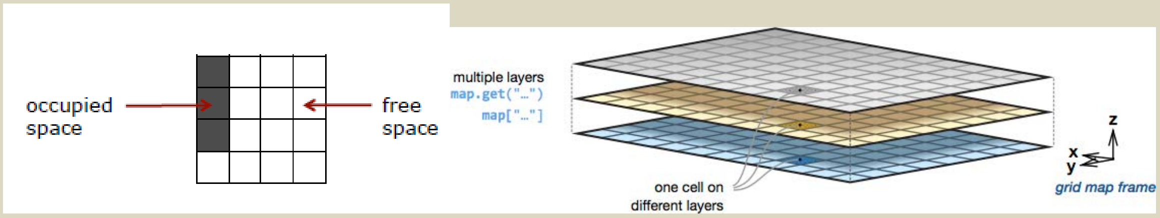
For the data collection, we would like to use the data collecting by the Autonomous Space Robotics Lab. The Canadian Planetary Emulation Terrain 3D Mapping Dataset is a collection of three-dimensional laser scans gathered at two unique planetary analogue rover test facilities in Canada. These test facilities offer emulated planetary terrain in controlled environments, as well as at manageable scales for algorithmic development. This dataset is subdivided into four individual subsets, gathered using panning laser rangefinders mounted on mobile rover platforms. This data should be of interest to field robotics researchers developing algorithms for laser-based Simultaneous Localization And Mapping (SLAM) of three-dimensional, unstructured, natural terrain. All of the data are presented in human-readable text files, and are accompanied by Matlab parsing scripts to facilitate use thereof. (http://asrl.utias.utoronto.ca/datasets/3dmap/)

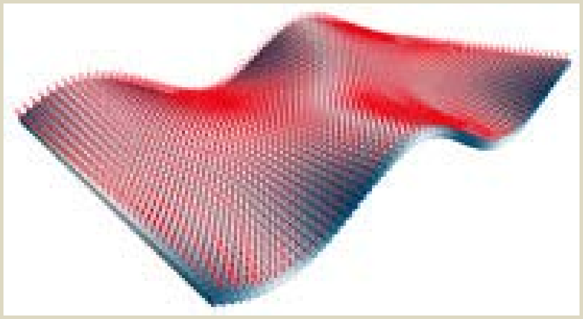
Otherwise, Autonomous Space Robotics Lab has build other similar dataset which using the lidar as sensor to collect the environment data. The Gravel Pit Lidar Intensity Imagery Dataset is a collection of 77,754 high-framerate laser range and intensity images gathered at a suitable planetary analogue environment in Sudbury, Ontario, Canada. The data were collected during a visual teach and repeat experiment in which a 1.1km route was taught and then autonomously re-traversed (i.e., the robot drove in its own tracks) every 2-3 hours for 25 hours. The dataset is subdivided into the individual 1.1km traversals of the same route, at varying times of day (ranging from full sunlight to full darkness). This data should be of interest to researchers who develop algorithms for visual odometry, simultaneous localization and mapping (SLAM) or place recognition in three-dimensional, unstructured and natural environments.

Experimental Design and Methods

Software Prepare

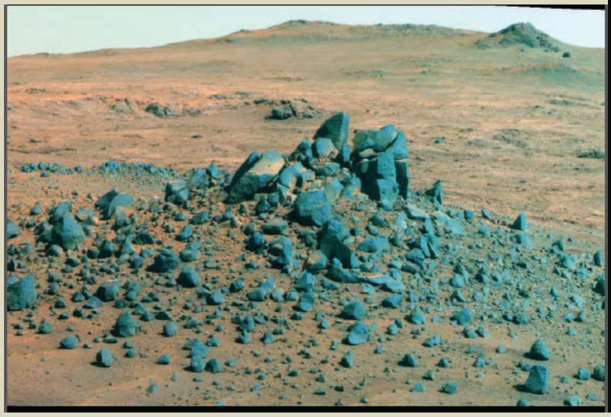
We would use the Grid Map library as major software, Grid Map is a C++ library with ROS interface to manage two-dimensional grid maps with multiple data layers. It is designed for mobile robotic mapping to store data such as elevation, variance, color, friction coefficient, foothold quality, surface normal, traversability etc. We will use it in our project to do the elevation mapping. The library provide lots package or functions. We could use image_to_gridmap_demo to convert data from an image to a grid map. Then, filters_demo uses a chain of ROS Filters to process a grid map. Starting from the elevation of a terrain map, the demo uses several filters to show how to compute surface normals, use inpainting to fill holes, smoothen/blur the map, and use math expressions to detect edges, compute roughness and traversability. The principle of Occupancy grid mapping is more about the mathematical probability problem. First, the algorithm divides the image to a lot of grid cells, each grid cell is rigid. We use these cells to represent our environment. Each cell is described as two values, occupied or free space. Then, combine the cells from different layers, we could get the depth of every single space position by some mathematical way. Once get the depth of the space position from the sensor position, the algorithm could convert it to the real space places and map it. Grid map is an powerful open source library, however, figuring out how to use is difficult. (https://github.com/ethz-asl/grid_map)





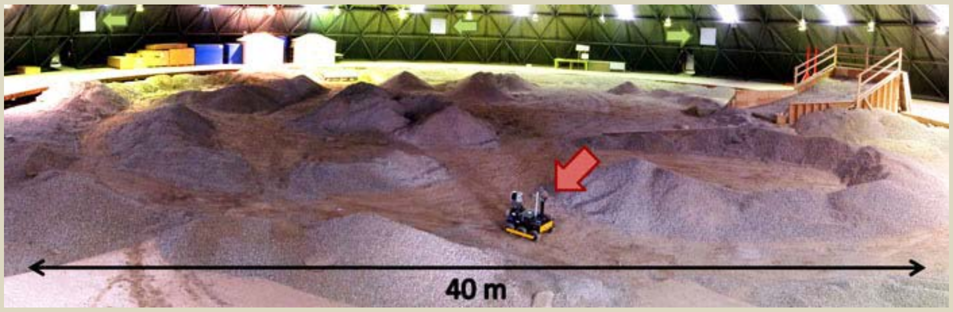
There are many researches about the indoor 3D mapping by using Grid Mapping approach. Outdoor unknown local environment mapping is similar but still have a lot of differenties. Mars surface is rocky and out-of-flatness, which are seen as nosies. Noises would affect the computing process, computer need long time to decide the noises and deal with it. Because the depth of each grid cell partially depending by the nearby cell, noises may affect lots.

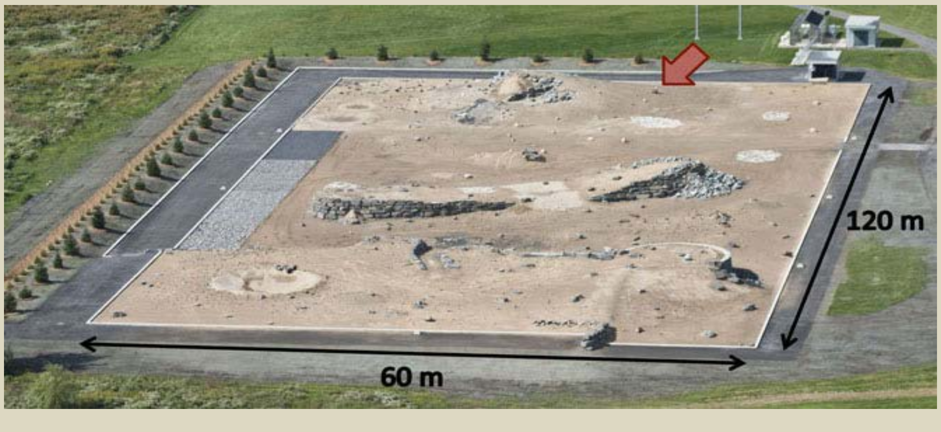
The most difficulties here is we cannot assume our robot is always in the certain position. Because of rocky ground, the data or image from two or more lenses are often in the non-horizontal positions, how to get the continues certain elevating map with uncertain position of camera is a critical question we want to deal with.



Data prepare and implement

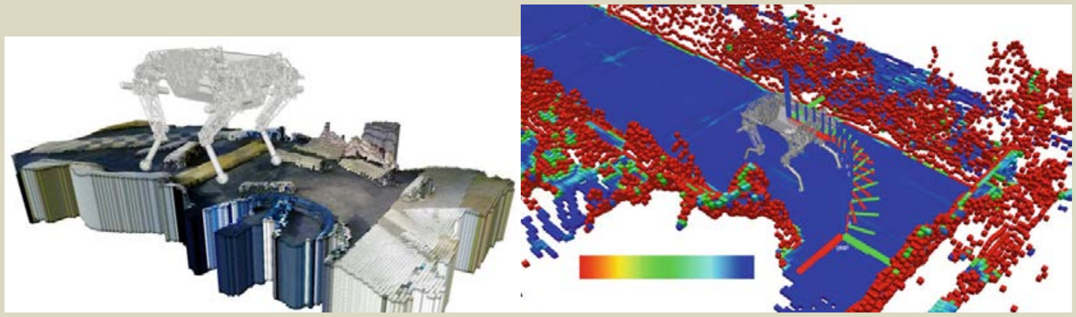
We would use the data collecting from the Autonomous Space Robotics Lab. Their robot was navigating in a planetary emulation terrain. They have the datasets from the indoor experiment and the outdoor experiment. We would try the indoor data first since it has few noises and much easier for us to familiar with the dataset. The dataset provides the stereo imagery, sun vectors, and inclinometer data. Otherwise, they have also used the laser as sensor. These data are important for us to do the 3D mapping. To using the data for 3D mapping, we need find the way to convert the data we got, take videos as an instance, to the data style that our software can be used. (http://asrl.utias.utoronto.ca/datasets/devon-island-rover-navigation/rover-traverse.html#Top, http://asrl.utias.utoronto.ca/datasets/3dmap/)





Uncertainty and noise handling

We will use the research that Fankhauser et. al. did in 2014. They came up with an algorithm to do the elevation mapping with the uncertainty position of the legged robot. This algorithm can explicitly handle drift of the robot pose estimation which occurs for many autonomous robots. Their mapping approach fully incorporates the distance sensor measurement uncertainties and the six-dimensional pose covariance of the robot. The algorithm is already defined in the Grid Mapping library.



We checked many works done by others. However, most of them are not work because of being outdated, specific instruments, or unknown reasons.

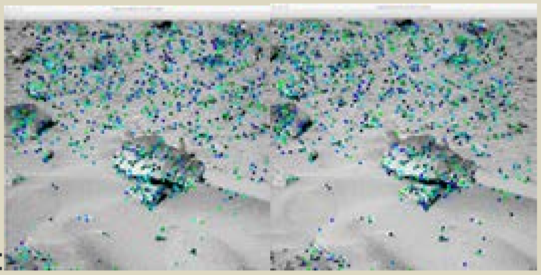
Result

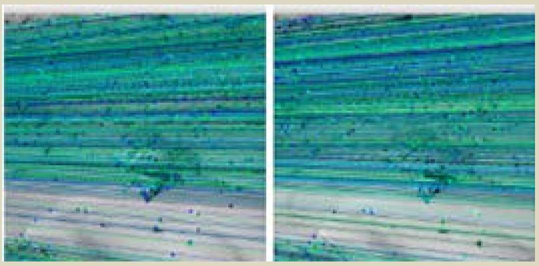
The first question we need to deal with in the research is about how to 3d map the image. We tried modules from OpenCV to see if the pair of stereo images could be used efficiently.

Input:

Left: Right:

Output:

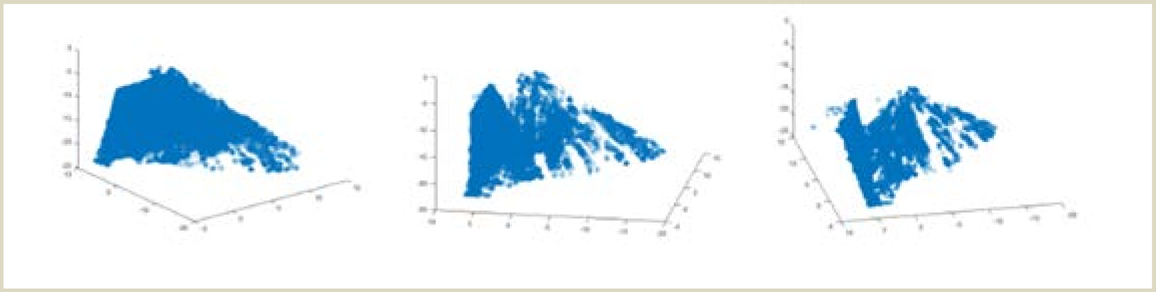
SURF module: 

SIFT module: 

Results: we could see that the module could find the similarities between two images, which means this pair of stereo image could be used for next 3d mapping by using opencv package.

Then, we use this pair of images to develop the 3D picture by using module from the opencv.

Output:



Results: For roughly mapping, the result is fine, the module could find there is a slope showed in images. However, for robot running, the results show that the module from openCV is not working very well for mars environment 3d mapping. For Mars rover running on Mars, the mapping should be precise, but this module could get the precise map.

Reference

Se, et. al. “Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks”

Ali, Nasser H., et. al. “Kalman Filter Tracking” *International Journal of Computer Applications (0975 – 8887)* Volume 89 – No 9, March 2014

P. Fankhauser and M. Hutter, "A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation", in Robot Operating System (ROS) – The Complete Reference (Volume 1), A. Koubaa (Ed.), Springer, 2016.

M. Don and L. James J. “Using Real-Time Stereo Vision for Mobile Robot Navigation” Autonomous Space Robotics Lab. “Canadian Planetary Emulation Terrain 3D Mapping Dataset”. http://asrl.utias.utoronto.ca/

Autonomous Space Robotics Lab. “Devon Island Rover Navigation Dataset”

Fankhauser, Péter et. al. “A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation

Jet Propulsion Lab. “In-situ Exploration and Sample Return: Autonomous Planetary Mobility” https://mars.nasa.gov/mer/technology/is_autonomous_mobility.html

P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Siegwart, "Robot-Centric Elevation Mapping with Uncertainty Estimates", in International Conference on Climbing and Walking Robots (CLAWAR), 2014.

Rusu, Radu B. and Steve C.“3D is here: Point Cloud Library (PCL)”

Eigen.tuxfamily.org. Eigen. http://eigen.tuxfamily.org/index.php?title=Main_Page

Se, Stephen and Little, David Lowe Jim. “Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks”.