ARISTOTLE UNIVERSITY OF THESSALONIKI

SLAM for Autonomous Planetary Exploration using Global Map Matching

Author:
Dimitrios GEROMICHALOS

Supervisor:
Assoc. Prof. Loukas PETROU

A thesis submitted to the Faculty of Engineering in partial fulfillment of the requirements for the degree of

Diploma
in
Electrical and Computer Engineering

Thessaloniki,
February 2018

Abstract

Write abstract here...

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Mention the scope, its state, the current issues and what is needed to solve them

- Scope: planetary exploration (for scientific purposes)
- State: past and current missions, their purpose and outcome/state
- Issues: no constant communication, manual command sequence, unsafe/upredictable conditions
- Needed: real-time and low-supervision system/platform

1.2 Problem Statement

Mention the environment and the robot's equipment and purpose
Mention that the problem is given X inputs to calculate Y outputs

- Environment: extreme planetary terrains with anomalies (rocks and craters) and high elevation variance
- Equipment: mechanical and sensor configuration
- Purpose: autonomous (low-supervision) navigation
- Inputs: odometry pose, sensor inputs, imu, global map
- Outputs: corrected pose, local elevation map

1.2.1 Presumptions

Explain what presumptions are made to the problem

- Initial global position is known
- Environment is static (not true for autonomous driving applications of algorithm)

- Environment is unknown (no high resolution a priori maps) (not true for autonomous driving applications of algorithm)
- Terrain is single layered (no bridges etc.)

1.3 Literature Review

1.3.1 SLAM

- What is SLAM
  - typical explanation in literature
  - categories
- How is SLAM implemented
  - volumetric/feature based approaches
  - grid-based fast slam
  - etc. etc. etc.

1.3.2 Planetary Absolute Localization

Mention how is absolute localization achieved in planetary applications

- Feature based approaches
- Skyline based approaches
- map based approaches

1.4 Thesis Objectives and Outline

1.4.1 Research Objectives

Mention what are the main objectives in terms of research

- Develop a novel technique for minimizing drift in global localization with global map matching and an execution strategy (criteria)
- Determine under what circumstances (i.e. resolution, local and global) can the orbital imagery be useful for SLAM
- Examine and quantify the gains (better localization, by providing a navigable map with the resolution and dense distribution restrictions that come with it) and loses (processing overhead)

1.4.2 Outline

Explain how this thesis is structured

Chapter 2

System Architecture

2.1 System Overview

In order for planetary robots to navigate autonomously in extreme and uncertain conditions where no GPS information is available, they need to perceive their surroundings with high precision and maintain this precision over time. A way of compensating for the lack of real-time GPS data, is to use an *a priori* global map that contains 3D information about the environment the robot is trying to navigate into. This map can be reconstructed using imagery taken from an orbiter and has a notably lower resolution compared to the robot's sensory data. To take advantage of this existing information, we have developed a system which is based around two layers of data processing.

In the first tier, the main objective is to capture the 3D structure of the environment using odometry, point cloud and IMU data from the robot's sensors. This is mainly achieved by adopting SLAM techniques to construct a local map and estimate the robot's current pose w.r.t. its initial pose. The latter, also called relative localization, is subject to drift in long-range traverses since all the input information is coming directly from the robot itself.

In the second tier, the objective is to achieve absolute localization by eliminating the accummulated position error of the previous stage. This is done by utilizing a matching technique to find the position of the robot's local map inside a global map.

The separation of the system to two layers comes from the need to process the data in separate moments, depending on the conditions of the environment as well as on the robot's processing capabilities. In Figure 2.1, the high-level design diagram of the system shows the 4 submodules that comprise the entire system. Their functionality will be explained in the following sections of this chapter.

2.2 Data Registration

An integral aspect of a robotic system, is the mapping process. The map is the model of the environment and can represent information about it in either two or three dimensions. In addition, the information a map represents can be organized either in a grid-based manner (volumetric map) or in the form of landmarks (feature-based

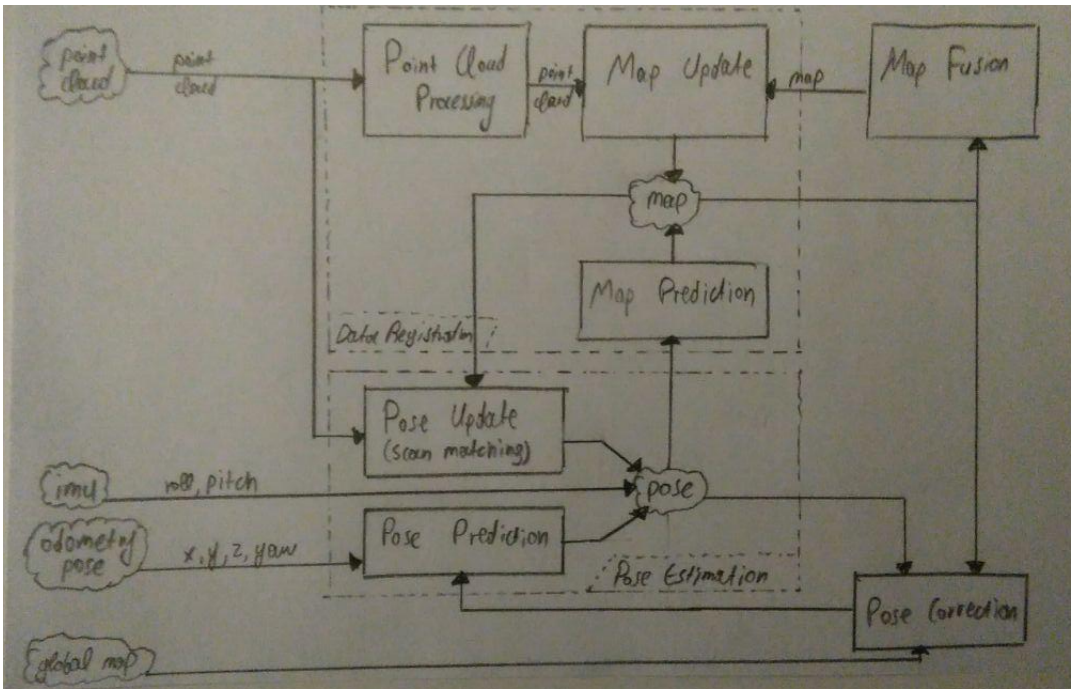

FIGURE 2.1: The high-level design diagram showing the submodules that comprise the system as well as the data flow.

map). Volumetric maps provide a more detailed and precise model of the environment compared to feature-based maps and are, therefore, the better choice for robots that aim to solve navigation tasks. Moreover, autonomous robots that need to navigate in non-flat surfaces, require a 3D model in order to safely avoid the terrain's obstacles. Robotic applications with this requirement include outdoor, underwater, airborne or planetary missions.

As we have already mentioned, planetary rovers have an important constraint in on-board computational power. This limitation deems the use of pure 3D maps inappropriate. To overcome this problem, we can represent the environment with a 2.5D map, known as elevation map. An elevation map is essentially a 2D grid-map composed of cells that hold the value of the terrain's elevation. It is important to note that this mapping method does not model completely the 3D environment, but can provide a sufficient approximation even for the most demanding applications.

To make use of this mapping technique, a robot must be equipped with a sensor that is capable of providing measurements in the 3D space. Such measurements are usually stored in a structure called point cloud - a set of points P that contain information about their position in space (x, y, z). A point cloud can be constructed from sensors that provide depth information, with the most popular being stereo cameras, time-of-flight (ToF) cameras and LiDARs.

In this section, the functionality of the Data Registration module will be explained. Its main purpose is to preprocess the input sensor data (point cloud) and register it to create an elevation map or update an existing one.

2.2.1 Point Cloud Preprocessing

The importance of preprocessing a point cloud before registering its data in a map, comes from the fact that point clouds contain large amount of points and hence process them is a time-consuming task. Additionally, the points represented in the cloud may be referenced in a different frame (usually the sensor's frame) than the frame of the map.

For this purpose, we implemented a point cloud preprocessing submodule, that consists of the following four steps:

1. **Voxelization**: In this step, we reduce the volume of the point cloud by downsampling it using a discrete 3D grid. Each cell of the grid, called *voxel*, is mapped to $\{0, 1\}$ and it therefore represents the existence or absence of an obstacle in that position. This method is called *voxelization* and results in a more sparse cloud of 3D points.

2. **Transformation**: A point P sampled from the sensor is in the sensor's reference frame S and need to be transformed in the map's frame M in order to be registered. This is be achieved by

$$\mathbf{t}_{MP} = \mathbf{R}_{SM}\mathbf{t}_{SP} - \mathbf{t}_{SM} \tag{2.1}$$

   where $\mathbf{t}_{MP}$ is the position of P in the map frame, $\mathbf{t}_{SP}$ is the position of P in the sensor frame, $\mathbf{R}_{SM}$ is the rotation between the sensor frame and the map frame and $\mathbf{t}_{SM}$ is the translation between them.

3. **Cropping**: Since a point cloud can span a wide area in space depending on the sensor's field of view and orientation, we can further reduce the volume of the cloud by discarding (cropping) points that fall out of the map. This is done by defining a box in space and removing from the point set all the points that are not inside it. In our case, we define the minimum and maximum cutoff points that represent the diagonal of the crop box as

$$\begin{aligned} min\_cutoff\_point &= \begin{pmatrix} -\frac{l_x}{2} - t_x, & -\frac{l_y}{2} - t_y, & z_{min} \end{pmatrix} \\ max\_cutoff\_point &= \begin{pmatrix} \frac{l_x}{2} + t_x, & \frac{l_y}{2} + t_y, & z_{max} \end{pmatrix} \end{aligned} \tag{2.2}$$

   where l_x and l_y are the lengths of the map in meters in the x and y axis, t_x and t_y are the positions of the map in meters in the x and y axis and z_{min} and z_{max} are the minimum and maximum elevation of the map.

4. **Uncertainty calculation**: In this last step, we use the sensor's model to calculate the variance of each point in the point cloud, σ_z^2. As a result, each point will contain in addition to a x and y position, a one-dimensional Gaussian distribution for the height. This calculation is different for every sensor, since it

depends on the sensor's characteristic. For a stereo camera, we approximate the variance as

$$\sigma_z^2 = \dots \tag{2.3}$$

2.2.2 Registration

Structure of Elevation Map

The elevation map, also referred to as *local map* or simply *map*, is a robot-centric grid-based map. This means that it is centered in the robot's position in the global reference frame. When the robot moves, previous cells that now fall out of the map are reset and values that belong to the new explored area take their place.

The map models the elevation of the environment as well as the uncertainty of it. As a result, in every position of the 2D grid, the elevation is represented by a one-dimensional Gaussian distribution. This means that the the map consists of two layers

- the layer of *mean* elevation (μ_z) and
- the layer of elevation *variance* (σ_z^2)

Apart from its structure, a map is also characterized by a few other properties. The most important are

- the *resolution*, which is the dimension of a cell in meters,
- the *length*, which is the length of the map in meters,
- the *minimum* and *maximum* elevation values that the map can hold,
- the 2D *position* of the map in the global reference frame and
- the *orientation* of the map in the global reference frame

Regarding the last property, it is worth mentioning that the orientation of the map is fixed w.r.t. to the global frame. This decision was made in order to reduce the complexity when performing a prediction of the map (see next section).

Map Prediction and Update

Even though the map prediction and update steps are grouped in the same submodule, they implement two separate functionalities. The map prediction is triggered when a new odometry pose is received and processed by the Pose Estimation module (Section 2.4). The map update is triggered as soon as a new point cloud arrives from the sensor.

- **Prediction**
  This step is responsible for moving the map when the robot assumes a new position. This is essentially a simple 2D transformation of the map's position in the x and y axes. The robot's orientation is not taken into account, since as we said earlier (Section 2.2.2), the orientation of the map about the z axis is fixed. To perform the prediction, we simply

  1. apply the delta translation (t_x, t_y) of the new estimated pose to map and
  2. keep the height Gaussian (μ_z, σ_z^2) of each cell unchanged

  When a translation is applied to the map, cells that fall out of the map have their values reset and elevation values of the new explored area take their place instead. To avoid unnecessary data copying in memory, the data storage can be implemented as a two-dimensional circular buffer.

- **Update**
  This is the core step of the Data Registration module, since this step implements the registration of the input point cloud into the map. The algorithm of the point cloud projection is shown in Algorithm 1.

  It uses a probabilistic approach to take advantage of the fact that both the point cloud (Section 2.2.1) and the map (Section 2.2.2) represent the elevation as a one-dimensional Gaussian distribution. The last operation of the algorithm (Operation 20) is the actual fusion, and can be implemented with different methods. A popular method is the one used in a Kalman filter, which given two Gaussian distributions $N(\mu_1, \sigma_1^2)$ and $N(\mu_2, \sigma_2^2)$, can be implemented as

$$innovation = \mu_2 - \mu_1 \tag{2.4}$$

$$gain = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \tag{2.5}$$

$$\mu_1 = \mu_1 + gain \cdot innovation \tag{2.6}$$

$$\sigma_1^2 = (1 - gain) \cdot \sigma_1^2 \tag{2.7}$$

2.3 Data Fusion

The goal of this module is to fuse data from different sources and increase the overall quality of the map. This is achieved by taking advantage the already computed uncertainty estimates in the map and applying fusion techniques to improve the existing data. In particular, we developed two submodules to perform fusion in separate stages of the pipeline. These are explained in the following sections.

Algorithm 1 Registration of point cloud to map

```
1:  map: two-dimensional grid of {μ_z, σ_z^2}
2:  point_cloud: set of {x, y, z}
3:  point_cloud_variances: set of {σ_z^2}
4:
5:  for every point in the point_cloud do
6:      if point is outside of the map then
7:          continue
8:
9:      point_mean ← point[z]                          ▷ projection of the point
10:     point_variance ← point_cloud_variances[point_index]
11:
12:     index ← find_map_index(point[x], point[y])     ▷ corresponding cell in grid
13:     cell_mean ← map[mean][index]
14:     cell_variance ← map[variance][index]
15:
16:     if cell_mean or cell_variance is not initialized then
17:         cell_mean ← point_mean
18:         cell_variance ← point_variance
19:     else
20:         fuse(cell_mean, cell_variance, point_mean, point_variance)
21:
22:     map[mean][index] ← cell_mean
23:     map[variance][index] ← cell_variance
```

2.3.1 Sensor Fusion

Using sensor fusion we can exploit the multiple sensors equipped in a robot to increase the covered area in the map. In addition to increased coverage, fusing sensor data that are overlapping in the map can increase its quality.

The way a point cloud is registered (fused) in the map, was already mentioned in a previous section (Section 2.2.2). Using this method we can combine information from different sensors, as long as the input data is in the form of a point cloud, by simply updating the map and taking advantage of the uncertainty estimates that are calculated in the preprocessing of the point cloud.

A drawback of this technique is that large calibration errors and inaccuracies in the transformations ($\mathbf{t}_{SM}$ and $\mathbf{R}_{SM}$) between a sensor and the map can have the opposite effect i.e. deteriorate the map quality. This is due to the fact that fusion can over-smooth obstacles since their correspondence in the map will be slightly different for each sensor. Errors like these can easily produce false negative obstacles and have a detrimental effect in the navigation of the robot.

2.4 Pose Estimation

Explain what is the purpose of this step Explain the particle filter used Explain how IMU measurements are fused to get the final pose

- Purpose: to get a better estimate of the pose using the local map
- Particle Filter: continuous, estimates 3 states (x, y, yaw) etc.

Add figure showing the particle distribution around the robot

2.4.1 Initialization

Explain the step

1. Sample all particles at initial pose

Add gaussian sampling equation

2.4.2 Prediction

Explain the steps

1. Add gaussian noise to delta pose transformation from visual odometry
2. Apply transformation to each particle

Add prediction equations (trivial)

2.4.3 Update

Explain the steps

1. Crop local map to sensor's dimensions
2. Create point cloud from local map
3. Crop raw (input) cloud to map size
4. Downsample raw and map point clouds (for faster convergence of point cloud alignment)
5. For each particle
   (a) Create particle point cloud by transforming map point cloud to particle's pose
   (b) Measure particle and raw point cloud alignment using mean square error
   (c) Assign inverse error as particle's weight

Add equations of point cloud alignment method for the weight update

2.4.4 Resampling

Explain the technique used, the steps and the strategy

- Technique: multinomial resampling
- Steps
    1. Normalize particle weights to form cummulative distribution
    2. Sample uniformally each particle from cummulative distribution
    3. Strategy: resample every X filter updates, depending on the robot's speed and the particle scattering (noise)

Add equations for forming and sampling from the cummulative distribution

2.4.5 Estimation

Explain the steps

1. Estimate pose from particle cloud:
    (a) Ignore particles with weights below a specific threshold
    (b) Pick top X particles
    (c) Calculate new estimate by performing a weighted average of particles
    (d) Calculate pose variance from the distribution of the particle cloud
2. Fuse IMU/Odometry states:
    (a) Get roll and pitch measurements directly from the IMU
    (b) Get z estimation directly from visual odometry
    (c) Fuse IMU (laser gyroscope) yaw measurement with the PF's yaw estimate by means of 1D gaussian fusion

Add equations for calculating the mean and the variance values of the estimate
Mention that the robot's absolute z estimate is not of importance and why is that

2.5 Pose Correction

Explain what is the purpose of this step
Explain the execution strategy of this step
Mention known limitations of this approach

- Purpose: to minimize absolute localization drift using the global map
- Strategy:

  - Generic: map needs to have features & robot must have covered certain amount of cells in the global map
  - Planetary specific: correct pose by the end of the day (for energy efficience reasons)

- Limitations:
  - When there are no features, the local to global map matching will fail
  - For safety purposes, the rover tries to generate a global path (using the global map) with as few features (i.e. rocks/craters) as possible

2.5.1 Global to Local Map Matching

Explain the steps
Explain why a template matching method is used instead of a feature-based one

1. Downsample local map to match global map's resolution
2. Calculate gradient (magnitude of the edge) of local and global map's elevation values
3. Replace unknown cell values with zero (does not affect matching)
4. Warp the local map to specific yaw angles to find a yaw correction
5. For each yaw angle: match local map to global map using template matching with normalized correlation coefficients Select or discard best match if score is above X (e.g. 95%)
6. Convert the matched position from image coordinates to map coordinates
7. Add gaussian noise to delta transformation (x & y) from matching
8. Apply transformation to each particle of particle filter

Add figures of:

- comparing Sobel edges with Scharr/Laplacian to justify the method
- source global/local maps in the scene (3D)
- source global/local maps in 2D
- downsampled local map
- gradients of global/local maps
- result image of matching with rectangles indicating the match & the ground truth match

Add equations of:

- nearest neighbor method for downsampling the local map
- gradient calculations (sobel kernel etc.)
- template matching method (normalized correlation coefficients)
- selection technique of best position from result image

2.5.2 Criteria Checking

Elevation Features Checking

Explain the steps and the threshold parameter selection

1. Create slope map from local elevation map by calculating the elevation gradient
2. Remove values in slope map below a specific threshold
3. Add remaining values to get absolute slope of map
4. Compare to threshold

Add equation for calculating the gradient and thresholding the map (image)
Add figures of final slope maps that succeed/fail the criterion

Traversed Distance Checking

Explain the step and the threshold parameter selection

1. Calculate traversed distance since last pose correction step
2. Compare to threshold

Add equation for the distance calculation (very trivial)
Add figures of robot traverses that succeed/fail the criterion

Chapter 3

System Implementation

3.1 Library

Mention GA SLAM library and tools used to create it etc.

3.1.1 Concurrency

Explain the algorithm's processing limitations and the circumstances under which it can run
Explain how concurrency can help and how it's implemented
Add timing diagram explaining the issues and how they are solved

3.1.2 Robotic Software Framework

Explain briefly ROCK, its tools, workflow and advantages/disadvantages
Explain briefly ROS etc.
Mention that the standalone library has interface to both frameworks

3.1.3 Orbiter Data Preprocessing

Explain that orbital imagery is emulated using drone imagery
Explain quickly how a 3D point cloud is reconstructed from the orbital imagery
Mention NASA's HiRISE (High Resolution Imaging Science Experiment)

Explain the steps for creating a global map from orbital point cloud:

1. Voxelize point cloud to the desired resolution
2. Crop the point cloud in the order of magnitude as local map's size
3. Smooth the point cloud using a SOR filter
4. Transform point cloud to robot's pose (using the initial absolute position)

Add figures of orbiter point cloud (raw & processed) and the orbiter map
Add figures from NASA's HiRISE depicting actual Mars point clouds

3.2 System Architecture

Explain how the algorithm can be integrated in a system and what components are needed (stereo, odometry etc.)
Add component diagram showing the architecture and how the different components are integrated

3.3 Planetary Rover Testbed

Explain lab's main rovers (HDPR & Exoter) and their mechanical/sensor configuration
Add figure depicting and explaining the rover(s)

Chapter 4

Experimental Validation

4.1 Scope of Experiments

Mention why mapping experiments are out of the scope
Explain briefly the purpose of the following experiments
Explain the type of the experiments

- Purpose: validate the algorithm and its robustness in various terrains and configurations
- Type: SIL tests (precollected data)

4.1.1 Environment

Explain the environment of the collected data (location, traversed paths etc.)
Add figures showing the said environment/traverses

4.1.2 Metrics

Mention the metrics used

- MSE of pose graph
- mean variance of pose graph,
- (execution times)
- etc.

4.2 Experiments on Pose Estimation

4.2.1 Relative Localization Results

Explain what the experiment is (test the accuracy of the particle filter)
Explain what is the expected accuracy of the estimation (1 cell size of local map)

Add bird's eye view of XY plot comparing:

- ground truth 2D position
- (Odometry)
- PF estimate

Add plot showing the error vs the number of particles used
Add plot showing the mean variance of the estimate vs the number of particles used
Add table showing the execution time vs the numner of particles used
Add plot showing the error vs the resampling frequency

Discuss the above results and what are the advantages/drawbacks while tuning each parameter

4.3 Experiments on Global Map Matching

4.3.1 Absolute Localization Results

Explain what the experiment is (test the drift correction on long range traverses)
Explain what is the expected accuracy of the correction (1 cell size of global map)

Add bird's eye view of XY plot comparing:

- ground truth 2D position
- (Odometry)
- PF estimate without global correction
- PF estimate with global correction

Repeat test in environment with different distribution of features
(Repeat test in environment without features to show the limitation of the approach)

Add table with the matching accuracy of the different experiments

Discuss the threshold value that should be chosen on such environments

4.3.2 Map Resolution Viability

Explain what the experiment is (test under what resolutions can we expect a drift correction)

Add figure showing local & global maps with different resolution
Add table comparing global vs local map resolutions using the correction error (actual offset - matched location)

Discuss the global and local map resolution of the current Mars missions and how these will change in the future (e.g. NASA is planning to have 0.25m orbiter map resolution by 2020)

Chapter 5

Conclusion

5.1 Thesis Summary

Mention briefly what was discussed in this thesis and what the experiments proved

5.2 Contributions

Mention the research and the development contribution of this thesis

- Research contributions (same as objectives of Introduction)
- Development contributions (system features):
  - Suitable for rough terrain navigation by utilizing 2.5D (elevation) maps
  - Global pose correction using map matching of local (robot-centric) and global (orbiter) elevation maps
  - Sensor-agnostic data registration using point clouds (support for lidar, stereo camera, ToF camera etc.)
  - Automatic sensor fusion when multiple inputs are provided (without prior configuration)
  - Adaptive design to fit the needs of different robots and applications

5.3 Directions for Future Extensions

Mention future work and possible extensions to the algorithm

- Use a discretized particle filter to gain advantage of the grid nature of the map
- Use global map for complete (all initial cells) or partial (new unknown cells at the map's edge) of the local map
- Use the motion model in addition to using the particle cloud distribution to determine the estimated pose's uncertainty
- Use a method (e.g. SOR) to minimize outliers in input point cloud

- Apply a threshold to points in projected point cloud that fall in the same map cell
- Implement a strategy for selecting a match in template matching when matches with similar score are a. close to each other b. far from each other

5.4 Applications

Talk about other possible applications except for the planetary one

- Autonomous driving vehicles in urban environment using a priori 3D reconstructed maps (i.e. Google Maps) - although GPS data helps in avoiding the drift of dead reckoning, global map can still be used for local map initialization
- Autonomous driving in rough non-urban areas were no predefined paths exists and GPS usage is limited
- Lunar teleoperated robot with SLAM for augmenting the operator's perception - processing is done on the station (i.e. ISS) since stereo images are sent anyway