SLAM for Autonomous Planetary Rovers with Global Localization

Dimitrios Geromichalos
Automation and Robotics Section
European Space Agency
Keplerlaan 1, 2201 AZ Noordwijk, Netherlands
Dimitris.Geromichalos@esa.int

Martin Azkarate*
Automation and Robotics Section
European Space Agency & University of Malaga
Keplerlaan 1, 2201 AZ Noordwijk, Netherlands
Martin.Azkarate@esa.int

Emmanouil Tsardoulias
Electrical and Computer Engineering Dept.
Aristotle University of Thessaloniki
54124, Thessaloniki, Greece
etsardou@eng.auth.gr

Levin Gerdes
Automation and Robotics Section
European Space Agency
Keplerlaan 1, 2201 AZ Noordwijk, Netherlands
Levin.Gerdes@esa.int

Loukas Petrou
Electrical and Computer Engineering Dept.
Aristotle University of Thessaloniki
54124, Thessaloniki, Greece
loukas@eng.auth.gr

Carlos Perez Del Pulgar
Systems Engineering and Automation Dept.
Universidad de Málaga
C/Ortiz Ramos s/n, 29071 Málaga, Spain
carlosperez@uma.es

Abstract

This paper describes a novel approach to SLAM techniques applied to the autonomous planetary rover exploration scenario in order to reduce both the relative and absolute localization errors, using two well proven techniques: particle filters and scan matching. Continuous relative localization is improved by matching high resolution sensor scans to the online created local map. Additionally, in order to avoid issues with drifting localization, absolute localization is globally corrected at discrete times, according to predefined event criteria, by matching the current local map to the orbiter's global map. The resolutions of local and global maps can be appropriately chosen for computation and accuracy purposes. Further, the online generated local map, of the form of a structured elevation grid map, can also be used to evaluate the traversability of the surrounding environment and allow for continuous navigation. The objective of this work is to support long-range low-supervision planetary exploration. The implemented SLAM technique has been validated with a dataset acquired during a field test campaign performed at the Teide Volcano on the island of Tenerife, representative of a Mars/Moon exploration scenario.

*Corresponding author

1 Introduction

1.1 Scope

Rover autonomy is of paramount importance in order to increase the scientific return of planetary exploration missions. To plan and command rover activities remotely can be a slow task. Operators' judgment of the risks and accordingly planning rover tasks is limited by the available information, i.e., the received rover telemetry such as camera images, surrounding panoramas or short range Digital Elevation Maps (DEM) and orbital imagery (Azkarate et al., 2016). Without certain onboard autonomous capabilities rovers may be hindered from reaching the scientific targets of their missions. This fact becomes more relevant as future planetary exploration missions are designed with more ambitious requirements in terms of distance to be traversed or number of robotic tasks to be performed. Take the example of missions for the Mars Sample Return Programme, where the Mars2020 rover (or a later Sample Fetching Rover) shall traverse around 10 to 20 km in one year and cache (or fetch) more than 20 samples within that period of time. In contrast, the average traversed distances of previous Mars rover missions ranged from 1 to 4 km per year, during which rovers have performed less complex robotics tasks (Jet Propulsion Lab, 2018a; Jet Propulsion Lab, 2018b; Vago, 2017; Wilson, 2013; Merlo et al., 2013).

In this paper we address the rover autonomy from the navigation perspective. Rovers autonomously navigating on a planetary surface require several software modules and on-board capabilities that allow them to assess the surrounding risks, plan a safe path to the next scientific target and then have a controlled traversal of that planned path. In particular, mapping processes become necessary in order to evaluate the unknown hazards of the terrain ahead and self-localization is required for computing a safe path and controlling its execution. SLAM techniques are commonly employed in mobile robotics applications for solving these two problems, i.e., mapping and localization.

According to the survey in (Jefferies and Yeap, 2008), and adopting the categorization and nomenclature defined therein, an autonomous planetary rover would require an online and volumetric SLAM method to fulfill its navigation tasks and could also safely assume a static environment. In order to solve the SLAM problem, i.e., generating a model of the environment (a map) and keeping track of the trajectory within that model, numerous approaches have surfaced during the past years in the robotics and computer vision (CV) community, which mainly include probabilistic, optimization, and scan matching based approaches.

In this paper we propose the combined use of scan matching and probabilistic techniques to provide the rover with the necessary means to support long-range autonomous navigation. The main contributions of this paper are an efficient design and implementation of a SLAM technique targeting the specific needs of a planetary rover and an additional process that complements the SLAM in order to correct the localization error in the global reference frame. Moreover, the proposed approach has been implemented and experimentally validated. For this validation a dataset gathered during a field test campaign has been used, running a representative planetary rover on an analogue terrain.

1.2 Related Work

1.2.1 SLAM

The Defense Advanced Research Projects Agency (DARPA) Grand Challenges of 2004 and 2005 and the 2007 DARPA Urban Challenge sparked the research interest in the field of autonomous driving. The goal of these challenges was to navigate autonomously in desert trails and urban environments spanning 244km and 97km respectively. It became obvious early in the first challenge that GPS alone is insufficient for vehicle localization, as the accuracy requirements were in the order of decimeters. The following challenges, however, demonstrated autonomous navigation capabilities in large scale for the first time. An important ingredient of the teams that finished the challenges successfully was the use of SLAM for creating offline landmark

maps used for the vehicle to localize against(Thrun et al., 2006).

Two of the most common probabilistic techniques in modern robotics are Particle Filters (Murphy, 1999; Doucet et al., 2001) and Extended Kalman Filter (EKF) (Thrun et al., 2004). A particle filter is in practice a Bayesian inference method that estimates the state of a dynamic system from controls and sensor measurements, and has similar applications as the Kalman filter, but is better at handling large dimensionality (Thrun et al., 2005). In the early 2000s, particle filters became one of the most popular approaches of solving the SLAM problem. A relevant algorithm called FastSLAM (Montemerlo et al., 2002; Montemerlo and Thrun, 2003; Barfoot, 2005) uses a class of particle filters called Rao-Blackwellized particle filters (RBPFs) (Doucet et al., 2000; Murphy and Russell, 2001) and later work combined these filters with scan matching (Grisettiyz et al., 2005; Hahnel et al., 2003). In Rao-Blackwellized filters each particle contains a candidate map of the environment which is why effort was spent on optimizing, i.e. reducing, the number of particles needed to build a consistent map (Sim et al., 2005). Equally important was the effort in solving the issues of particle depletion using efficient data structures and data association that would allow for successful loop closures (Hahnel et al., 2003).

Scan matching is one of the oldest and simplest methods used to solve the SLAM problem using Occupancy Grid Maps (OGM), i.e., maps that represent the continuous space using a discrete uniform grid, and notably can be utilized for navigation purposes. In scan matching, the position of the robot is calculated by matching, i.e., computing the transformation between, the current and previous sensor scans. Based on this concept, numerous implementations and variations have been developed due to its simplicity (Rusinkiewicz and Levoy, 2001; Brenna, 2008).

Later work in the last decade has tried to solve the online SLAM problem using optimization techniques. The most relevant work in this area is known as Graph SLAM (Grisetti et al., 2010; Kümmerle et al., 2011). In Graph SLAM the robot poses are represented as a graph where the nodes correspond to the poses of the robot at different points in time, and the edges represent constraints between the poses (Thrun et al., 2005). The latter are obtained from observations of the environment or from movement actions carried out by the robot. The whole problem amounts to solving a large optimization problem, typically making use of the so-called Bundle Adjustment (Li et al., 2007). The graph can be shown to be a sparse graph of nonlinear constraints. Graph SLAM keeps track of all poses and measurements at current and previous times, providing a more consistent SLAM solution, both in terms of overall trajectory and environment representation, making it more robust against outliers compared to the aforementioned probabilistic approaches. To the authors' knowledge the state of the art in this approach can be found in (Mur-Artal and Tardós, 2017), which yields a very efficient solution for feature tracking and relative localization, exploiting the Oriented FAST and rotated BRIEF (ORB) (Rublee et al., 2011), a keypoint detector and feature descriptor that is widely accepted by the Computer Vision community. However, given the sparse nature of the produced map it is hard to assess its traversability, which is why it has rarely been used as an active SLAM solution, but rather as a passive one. This means that the produced map is not used to steer and navigate the rover as it moves.

Most of the mentioned related work in the area of SLAM has been produced within the scope of robotics for Earth applications. Due to the inherent constraints in computational power of space systems and their hard and costly qualification procedures we seldom find cases of actual space missions making use of these complex autonomous navigation solutions. Work in (Matthies et al., 2007; Maimone et al., 2007; Cheng et al., 2005) shows the first CV algorithms used by the Mars Exploration Rovers (MER) in order to achieve on-board relative localization using Visual Odometry (VO). But it is only in recent years and after important flight software updates that the MER and Curiosity rovers have operated autonomously during some traverses and only under certain circumstances (Maimone, 2017). While Europe has yet to launch its first rover to Mars (and Space), the European Space Agency (ESA), together with other space agencies and industrial partners, has been working for years on solutions for autonomous planetary exploration. Fully autonomous capabilities are not implemented for the ExoMars rover yet, however in (Shaw et al., 2013) the VO implementation that the rover will run is explained. The project SPARTAN (Kostavelis et al., 2011; Siozios et al., 2011; Kostavelis et al., 2014; Lentaris et al., 2015) and successive initiatives made significant effort towards optimizing and efficiently bringing localization and mapping processes to space representa-

tive FPGA hardware implementations. Additionally, during the last two decades the French Space Agency (CNES) has developed their solution for AutoNav (Moreno, 2013; Bousquet, 2011) which they have proved to work in both simulation and field experiments. Finally, one of the most relevant field test activities in a space representative environment and platform was performed during the SEEKER campaign (Woods et al., 2014) in the Atacama Desert, Chile, in which the rover managed to autonomously traverse a total of 5.05 km during one day. As of now however, all these space-related implementations have dealt with the localization and mapping processes separately, in sequential pipelines, similar to the strategy presented in (Correal and Pajares, 2011) and do not strictly fall in the category of SLAM.

Researchers in the robotics community have already tried to bring several SLAM approaches to the realm of space planetary exploration, motivated by optimizations that could render SLAM a viable solution to fit the space constraints. Latest work in (Hidalgo-Carrió et al., 2018) is probably the most remarkable effort in that direction. Its Adaptive SLAM approach, based on ORB, puts the stress on selecting the minimum amount of *keyframes* needed for the SLAM solution to converge in the back-end, using Gaussian Processes to model the residual error of a novel kinematic model approach for the odometer pose estimation (Hidalgo-Carrió et al., 2017). Eventually the number of processed frames can be reduced to 25 % with marginal increase on the localization error, making this a viable solution for online SLAM for planetary rovers. In addition, it also produces a dense reconstruction of the environment by stereo processing the selected *keyframes* which could be used for traversability estimation and navigation purposes.

1.2.2 Global Localization

It is also important to note that since these techniques only utilize sensors which are relative to the rover, drift is inherently built up in localization, such as in dead-reckoning techniques, and therefore reduce the quality of the SLAM solution. While previous works managed to build and keep a consistent map of the traversed environment and the robot pose in it by successfully identifying loop closures (Bampis et al., 2018), we must agree that rover missions do not revisit traversed areas on their daily routine. Therefore the chances of detecting any loop closure are scarce and still would require running long optimization processes over large windows of a pose graph that would be difficult to keep within the memory resources of the rover system. This makes the loop closure approach non-valid for our scenario, hence our challenge. On the other hand, the Mars exploration scenario offers certain *a priori* information that allows to tackle the issue of accumulated drift, and keep track of the rover's position in the global scale.

Several approaches for solving the global localization for planetary exploration have been researched in the past. Skyline-based solutions (Stein and Medioni, 1992; Cozman and Krotkov, 1998) were presented in the early 90s as a means of finding a coarse (in the order of hundred meters) localization of the rover from an unknown initial state. VIPER (Cozman et al., 2000; Chiodini et al., 2017) is a well known algorithm that matches the horizon skyline signature captured by a panoramic picture acquired by the rover, to predicted skyline signatures at various positions on the global map. Feature-based solutions later presented were capable of showing higher precision in global localization (up to a few meters). Feature-based approaches, in general, initially extract interest points from the global and local maps and then they match them in search of global-local feature correspondences. MOGA, presented in (Carle et al., 2010), showed the possibility to match features of 3D maps generated by LiDAR scans obtained by the rover with features of orbital elevation maps, and was successfully tested in a Mars-Moon analogue site on Devon Island, Nunavut. Rover generated maps were produced using the 3D SLAM approach (Tong et al., 2011; Tong et al., 2012). However, LiDARs are hardly used in planetary rovers due to the heavy weight and high power consumption and to the authors knowledge there exists no LiDAR sensor qualified for planetary rover space missions. Other notable work in (Hwangbo et al., 2009; Boukas et al., 2018) proposed solutions based on matching networks of landmarks or regions of interest that are traceable both from ground and orbital images. A global network is calculated offline from orbital images around the landing site ellipse, while the local network is calculated along the traverse. As soon as enough rocks or landmarks are introduced in the local network their distribution pattern can be matched between both networks.

1.3 Approach to SLAM for Planetary rovers

The work we present in this paper has the motivation of designing a SLAM implementation fit for the planetary rover exploration case, and in particular for Mars rovers. We focus on the specific needs and conditions that Mars exploration poses and solve the localization and mapping problem that we believe best allows for long-range autonomous navigation. Therefore, in order to design our SLAM algorithm efficiently, we first need to analyze the specific requirements and conditions of a planetary rover navigation scenario.

With regards to the mapping process, it is important to note that the on-board generated local map serves mainly for the extraction of a traversability grid that is needed for the local path planner or in general to avoid obstacles and steer the rover in the local area. There is typically no mission requirement for continuously generating onboard detailed 3D maps of the traversed terrain nor to keep a consistent and growing map in memory, which would eventually lead to generating a global traversed map. Note that the 3D reconstruction and mapping processes that are used in the interest of planetary geology science are later run on Ground with a selection of images downloaded from the rover telemetry. In subsection 1.2 we mentioned the two map representations that are popular in the literature, i.e., the landmark- or feature-based and volumetric, such as the occupancy- or elevation-grid-based. Volumetric grids are effective for dense ambiguous information while landmarks are better suited for sparse unique features. In order to obtain the traversability map, the on-board mapping process shall be focused on generating a volumetric elevation grid map (2.5D) where only the highest z value (elevation) for each (x, y) coordinate is estimated. This can easily be processed to detect obstacles and slopes and evaluate traversability. We stress that no dense 3D reconstruction is needed, just a structured grid with elevation estimates. Similarly, the required resolution of the grid map (size of grid cells) is also dependent on the locomotion capabilities of the rover, which are determined by the dimensions and characteristics of the rover platform and wheels. This way, the local map size and resolution can be optimized, significantly reducing the amount of memory needed to store the map. For example, if we define a map of

2 GA SLAM

Following the previously mentioned conditions and needs of a rover on a planetary exploration mission, we have designed a SLAM algorithm to achieve precise relative and global localization, and a local map usable for autonomous navigation, while trying to keep the computational overhead as low as possible. For mapping, we apply a two step process by initially filtering the input 3D sensor data to the necessary resolution and then updating the height value of each cell according to a Kalman-based fusion rule. Additionally, combining data from different camera sensors can help improve the outcome of the mapping process and increase the sensor coverage. For relative localization, a scan-to-map technique is used that implements a particle filter combined with the Iterative Closest Point (ICP) algorithm (Zhang, 1994). For global pose correction, we use template matching, a traditional CV technique, that enables us to match grid maps. Figure 1 shows the design diagram with the interfaces between the main modules. Pose Estimation follows a two step process. Pose prediction is done using odometry inputs and as soon as a new sensor point cloud is acquired the pose is updated. The updated pose is fed to the Data Registration module to perform the map prediction and the filtered point cloud can be used to update the map. Global Pose Correction is performed at discrete times using the latest available map, in combination with the global orbital map. In the following subsections, we describe the algorithm of the Globally Adapted SLAM (GA SLAM) for planetary missions in more detail.

2.1 Data Registration

The main objective of the mapping process is to obtain a local elevation map from which to derive the traversability of the terrain ahead.

As explained in subsection 1.3, for computational efficiency we want to optimize the designed process to

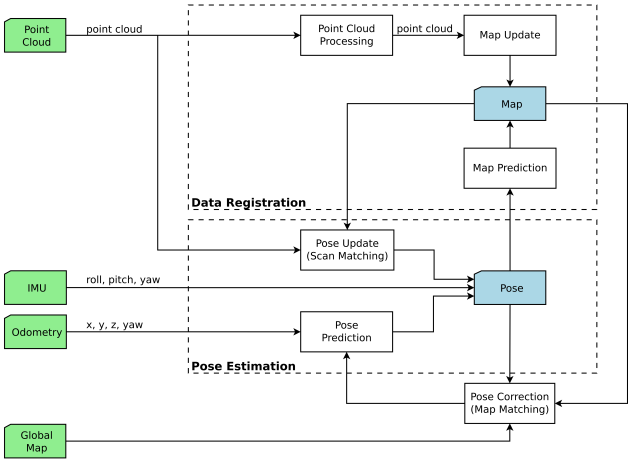


Figure 1: The high-level design diagram of the GA SLAM system.

solve the navigation task of a planetary mission, adapting the resolution of the produced volumetric maps. It is important to note that this mapping method does not completely model the 3D environment, but can provide sufficient information for the rover's navigation. It is also worth mentioning that the generated grid map will be used again in both the relative and absolute localization processes, which is relevant for the efficiency of the proposed SLAM approach.

The mapping process requires the rover to be equipped with sensors that are capable of providing measurements in the 3D space, such as stereo cameras, time-of-flight (ToF) cameras, or LiDARs. Regardless of the sensor, the input measurements are stored in the form of so-called structured point clouds, i.e., a set of points p that contain information about their position in space (x, y, z) (Rusu and Cousins, 2011). Before registering an input point cloud, we preprocess it with the aim to minimize the required computations. The preprocessing is a 4-step submodule that consists of:

1. **Voxelization**: Downsampling of the point cloud using a discrete 3D grid, resulting in a sparser cloud of 3D points (Hinks et al., 2013).
2. **Transformation**: Transforming from the sensor reference frame to the map's reference frame. This is achieved by:
$$t_{MP} = R_{SM} t_{SP} - t_{SM} \tag{1}$$
where t_{MP} is the position of P in the map frame, t_{SP} is the position of P in the sensor frame, R_{SM} is the rotation between the sensor frame and the map frame and t_{SM} is the translation between them.
3. **Cropping**: Cutting off the points that fall out of the area contained by the map.
4. **Uncertainty calculation**: Each point will contain – in addition to an (x, y) position – a one-dimensional Gaussian distribution $\{\mu_z, \sigma_z^2\}$ for the height. σ_z^2 is calculated by using the sensor model and propagating the uncertainty of the transformed point according to (Ochoa and Belongie, 2006). For the case of a stereo camera the sensor model in (Huesing et al., 2014) can be used to estimate σ as:
$$\sigma = \frac{c \cdot \tan(0.5 \cdot f)}{0.5 \cdot b \cdot w} d^2 \tag{2}$$
where c is the search range or disparity precision, f is the field of view, b is the baseline, w is the horizontal resolution of the camera and d is the range of the point.

After preprocessing, the point cloud data is ready to be registered in the local elevation map. This local map "follows" the rover, being always centered at the robot's position in the global reference frame, the

same reference frame as the one used to define the global map. When the robot moves, previous cells that now fall out of the map are reset and values that belong to the newly explored area take their place using a circular buffer. 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. The map therefore consists of the *mean* elevation layer $\{\mu_z\}$ and the elevation *variance* layer $\{\sigma_z^2\}$ and is also characterized by properties such as, resolution, length, elevation range, the 2D position in the global reference frame and the orientation in the global reference frame. It is worth mentioning that the orientation of the map is fixed and aligned with respect to the global reference frame, and therefore does not rotate with the heading of the rover. This decision was made with a focus on reducing the complexity when performing the map prediction step.

The mapping process is a continuous iteration between prediction and update steps. The former is triggered when new odometry information is received (see subsection 2.2) and the latter is triggered when a new point cloud arrives from the sensor:

- **Prediction**: Apply to the map the delta translation corresponding to the new pose estimation. This is essentially a simple 2D transformation of the map's position in the x and y axes. In this process the Gaussian distribution of the height in each cell stays unchanged. Cells that fall out of the map after this translation are reset and newly explored cells take their place.
- **Update**: This is the core step of the Data Registration module, since it implements the registration of the input point cloud into the map. It uses a probabilistic approach to take advantage of the fact that both the preprocessed point cloud and the map represent the elevation as a one-dimensional Gaussian distribution. When input data falls in cells that are not yet registered in the map, input values are directly set into those cells. When input data falls in an already mapped area this registration represents the fusion of two Gaussian distributions, and can be implemented with different methods. A popular method is the one used in a Kalman filter (Cremean and Murray, 2005), which given two Gaussian distributions $N(\mu_1, \sigma_1^2)$ and $N(\mu_2, \sigma_2^2)$, the new distribution $N(\mu, \sigma^2)$ can be obtained as:

$$\mu = \mu_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \cdot (\mu_2 - \mu_1) \tag{3}$$

$$\sigma^2 = (1 - \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}) \cdot \sigma_1^2 \tag{4}$$

An example of the result of the Data Registration process can be seen in Figure 2.

Finally, it is worth mentioning that the data registration process is reconfigurable to different map sizes and resolutions and is appropriate for sensor fusion where point clouds coming from different sensors can be processed to improve sensor coverage as well as the statistical quality and robustness of the map.

2.2 Relative Pose Estimation

The goal of pose estimation, also referred to as relative localization, is to accurately track the rover's pose within the map created in the Data Registration process. The pose is comprised of the rover's position and orientation. In our application, we assume that the rover's Inertial Measurement Unit (IMU) can provide accurate and non-drifting *roll* and *pitch* values (Clerc et al., 2009; Durrant et al., 2017) and therefore be removed from the state space of the pose estimation process. Additionally, due to the reduced local variation of absolute altitude over the local map and the fact that the z position of the rover can be extracted from the (x, y) coordinates over the global map, this variable is also obviated in the state space and provided directly from the VO input. Therefore, to reduce complexity and computational overhead of the particle filter, the rover state we are trying to estimate in this problem is defined as:

$$\mathbf{x} = [x \;\; y \;\; \theta]^\top \tag{5}$$

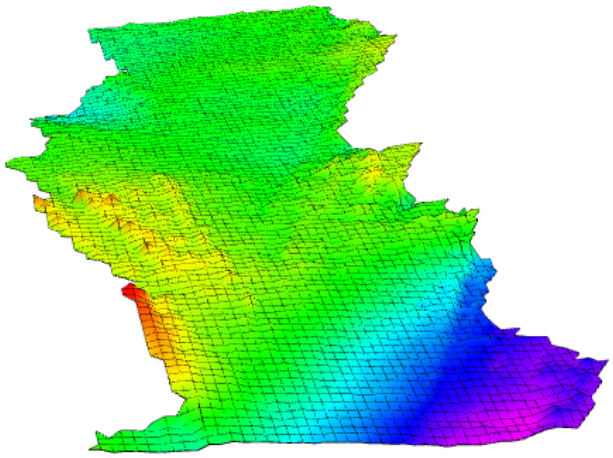


Figure 2: Perspective view of an example of the generated local elevation map after several data registration iterations. Colors encode the height of cells.

where θ is the orientation or *yaw* estimation. As already introduced in subsection 1.2 several methods exist to solve this problem. The approach we propose in this paper makes use of a combination of scan matching and particle filtering techniques. We employ a particle filter that samples particles in a continuous state space to estimate the posterior of the robot's state:

$$p(\mathbf{x}_T \mid m_{T-1},\ s_T,\ u_T) \tag{6}$$

where m is the constructed map of the environment, s are the sensory inputs, u are the odometry inputs and T is the current discrete time. This is done by using the odometry inputs to predict the state of each particle, and the sensory inputs, i.e. point clouds, for the weight update using scan matching. Finally, yaw measurements from the robot's IMU are fused with the final estimate to provide more accurate solutions.

The pose estimation algorithm can be summarized in the following steps:

- **Initialization**: This step is only executed at start of the algorithm execution or if the particle filter is reset and it consists of sampling the state of each particle (Equation 5) using a Gaussian distribution $\mathbf{X} \sim N(\mu, \sigma^2)$.
- **Prediction**: The state of each particle is transformed according to the delta pose transformation coming from the odometry input. At this step some Gaussian noise is also added to each particle. It is important to mention that the addition of noise to the robot's position is a simplification of the standard particle filter methodology, where noise is usually added to the robot's velocities.
- **Update**: When a new point cloud measurement s_t is received, the weight of each particle p_i is updated using scan matching. For every particle, the input point cloud is matched with a point cloud extracted from the robot's local map. Point clouds are preprocessed as explained in subsection 2.1 to achieve a faster execution. Since the weight of each particle is proportional to the uncertainty of its hypothesis, it can be updated by measuring the degree of alignment between the input point cloud and the particle's point cloud. This matching is computed by performing an ICP variant (Zhang, 1994). By running one single ICP iteration we obtain a fitness score that measures the alignment error of two point clouds as the sum of squared errors. The weight w_i of each particle p_i is then updated as:

  $$w_i = \frac{1}{\epsilon_i} \tag{7}$$

  where ϵ_i represents the ICP fitness score.

- **Resampling**: This step ensures that the particle population models the uncertainty of the problem as closely as possible (Douc and Cappé, 2005). A cumulative distribution is initially formed by normalizing the weight of each particle as

$$\hat{w_i} = \frac{w_i}{\sum_{k=0}^{N} w_k} \tag{8}$$

where w_i is the weight of a particle p_i from the particle set and $\hat{w_i}$ is the normalized weight. From the cumulative distribution, N new particles are sampled uniformly while maintaining the state of each particle. The high-weighted particles are more likely to be sampled and even repeated but the added Gaussian noise makes sure particles are again scattered. Different strategies for resampling can be followed. In our approach, we adopted a simple strategy that resamples every K iterations of the algorithm. This parameter can be configured accordingly depending on the magnitude of the Gaussian noise added and the rover controls.

- **Estimation**: Consists of extracting the final pose estimate from the particle filter. Simply picking the highest weighted particle could be an option, but this causes the pose estimate to "teleport" if the particle is not selected at the resampling step and produce discontinuities in the created map. To avoid this issue we follow the algorithm explained in Algorithm 1 which basically thresholds the particle cloud by their weight and then computes the weighted average of the top remaining clouds. Finally, to improve our estimate we fuse the yaw estimate with the IMU yaw measurement as two one- dimensional Gaussian distributions. For this we need to associate a variance to each of the variables. The IMU Gaussian is represented as $N(\mu_{imu}, \sigma^2_{imu})$ where μ_{imu} is the measurement value from the sensor and σ^2_{imu} is the variance which is a parameter that represents how reliable the sensor is. The second Gaussian is represented as $N(\mu_\theta, \sigma^2_\theta)$, where μ_θ is the yaw estimate that was extracted in the previous step and σ^2_θ is the variance of the distribution. It can be calculated by applying the formula for a weighted standard deviation as:

$$\sigma_\theta = \sqrt{\frac{\sum_{i=1}^{N} w_i(\theta_i - \mu_\theta)^2}{\sum_{i=1}^{N} w_i}} \tag{9}$$

where N is the number of particles, w_i is the weight of each particle and θ_i is the yaw hypothesis of each particle. The fusion of the two distributions is performed in a similar fashion as in subsection 2.1.

Algorithm 1 Estimation of final pose from particle population

P: set of particles
p: particle – set of states (x, y, θ) and an associated weight

$S \leftarrow \{\ \}$ ▷ candidate particles
for each p in P **do** ▷ remove uncertain particles
    if $p[w] > w_{threshold}$ **then**
        $S \leftarrow S + \mathrm{p}$

Sort S by weight
$M \leftarrow S[1:k]$ ▷ pick top k particles

$w_{sum} \leftarrow \sum_{i=1}^{k} w_i$
$x \leftarrow \sum_{i=1}^{k} w_i x_i \ / \ w_{sum}$ ▷ weighted average of each state
$y \leftarrow \sum_{i=1}^{k} w_i y_i \ / \ w_{sum}$
$\theta \leftarrow \sum_{i=1}^{k} w_i \theta_i \ / \ w_{sum}$

The extracted rover state estimate is again merged with the *roll* and *pitch* reading of the IMU and the z position coming from the VO to complete the rover's pose. Figure 3 shows a graphical representation of

the pose estimation filter. The figure shows the rover pose tracked along the trajectory, together with the current local map representation. Colors are used to represent the range of height values of the map and the cloud of red dots represent the particles of the filter around the rover estimated pose.

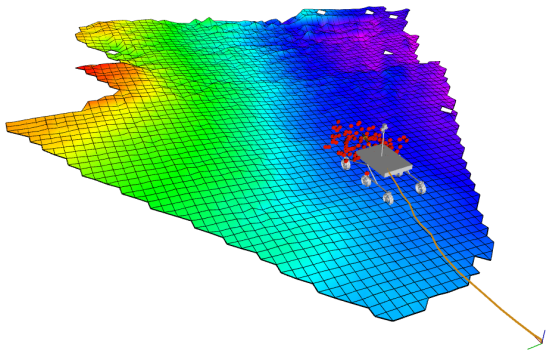


Figure 3: The particle cloud (red dots) of the particle filter distributed around the rover pose. The trajectory of the rover with respect to the initial state is represented by the orange line. The current local elevation map colors represent the range of height values from high (red) to low (purple).

2.3 Absolute Pose Correction

Although the rover's pose is constantly tracked using the particle filter in the Pose Estimation module, the approach will still build drift in localization, due to the fact that all the inputs of the process arrive from on board sensors. As a result, a localization error is accumulated in long-range traverses and the rover's pose in the global reference frame slowly drifts. With the aim of bounding this error and achieving absolute, i.e. global, localization, we have developed an additional component that eliminates the drift by matching the on board generated local map to a global one.

We assume the global map to be constructed from orbital imagery, and most certainly of a relatively lower resolution compared to the local map. Again, finding the match between the local and global map will correct the state of the rover's pose and consequently locate the rover in the absolute reference frame of the global map. Similarly, this step also corrects the 2D position of the local map but does not modify its content, i.e., the elevation values.

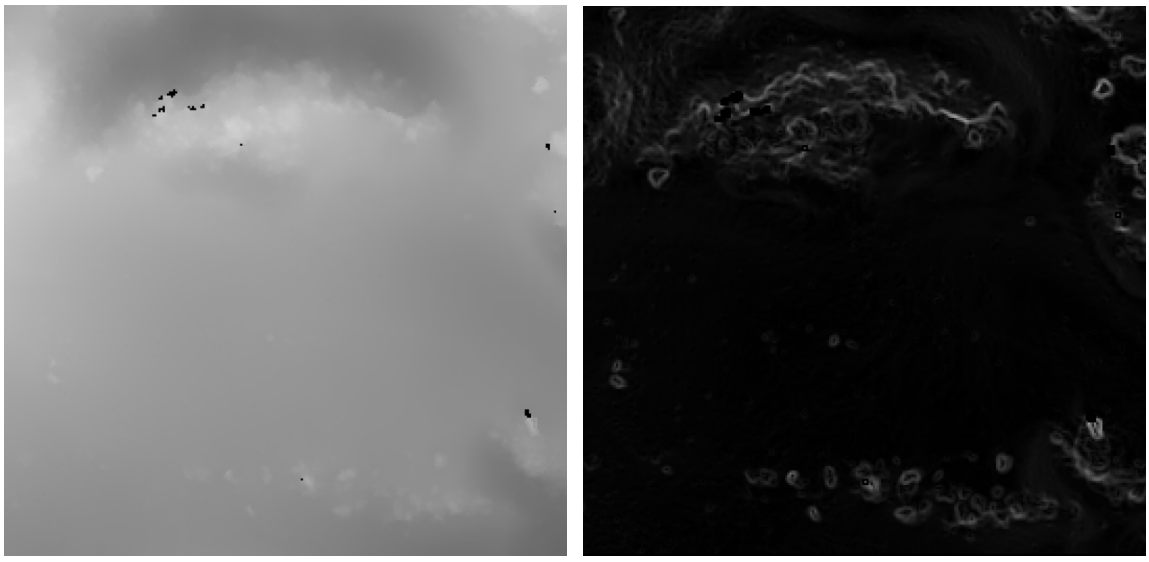
Due to the inherent differences in nature of the two maps, the pose correction module requires the preprocessing of them. First, the local map is downsampled to match the global map resolution and subsequently scaled up using a nearest neighbor interpolation method. Second, as it was explained in subsection 2.2, the z position of the rover state is directly taken from the VO input and merged at the particle filter output. In order to overcome the problem of possible long-term drift of the z estimation coming from odometry that could produce big differences between the local and global maps' elevation values, we compute the gradients of the two elevation maps. To obtain the gradients of the elevation maps we apply a Sobel operator over them. We calculate the gradients in x and y direction by convolving with a Sobel odd-sized kernel, which for a kernel size of three corresponds to:

$$K_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}; \quad K_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{10}$$

for the horizontal and vertical directions that provide the G_x and G_y gradients respectively. The final magnitude of the gradient is calculated by combining both gradients:

$$G = \sqrt{G_x^2 + G_y^2} \tag{11}$$

For example, in Figure 4 we show an elevation map and its correspondent gradient map, of an area 100×100 m^2 in size.



(a) Elevation image, 100×100 m^2 (b) Gradient image, 100×100 m^2

Figure 4: (a) The raw elevation image of the map. (b) Its gradient after applying the Sobel operator.

We approach the problem of local to global map matching as the matching of 2D images, since elevation and gradient grid maps can be interpreted as such (see Figure 4). Techniques for this vary from simple statistical-based to more complex template matching or feature matching (Jayanthi and Indu, 2016). Feature matching is usually the most efficient method in structured or feature-rich environments (e.g. Earth applications). However, the template matching approach is appropriate for our planetary scenario, also considering we are trying to find a correspondence of a smaller search image (template) into a bigger one (the source). Figure 5 illustrates how this search is done. A drawback of template matching compared to feature matching is that it cannot match rotated images, and therefore it cannot find a *yaw* correction in the robot's global pose. To overcome this limitation, we warp (rotate) the template image in a discrete angle space, e.g. from $-10\,^\circ$ to $10\,^\circ$ with a step of $1\,^\circ$. As a result, we run template matching for every individual angle (twenty times for the previous example) and choose the *yaw* correction angle that provides the best result.

Template matching works by sliding the template image T pixel-by-pixel in the source image I and comparing at each location the two images using a metric score. Then for each location of T over I, the score is stored in a result matrix R. Consequently, the location in R where the score is highest, is the location of the best match (Hashemi et al., 2016). From the different possible metric scores we choose to compute the Cross Correlation:

$$R(i,j) = \sum_{i',j'} \left(T(i',j') \cdot I(i+i', j+j') \right), \tag{12}$$

where x' and y' are indices describing the pixel locations in the template image whereas x and y are the indices of the pixels in the source image. This metric yields optimal results for our application where we may have unknown values of elevation in our local map. By normalizing the metric the output is in the fixed range of 0 to 1, with 1 being a perfect match. The matching threshold can be then fixed (e.g. 95 %) independently of the application.

Finally, the location of the template image where the metric is maximum and exceeds a certain threshold is used to correct the position of the rover's pose by applying the correction to all particles in the filter. The maximum obtained value of the Cross Correlation metric at that location is defined as the matching score.

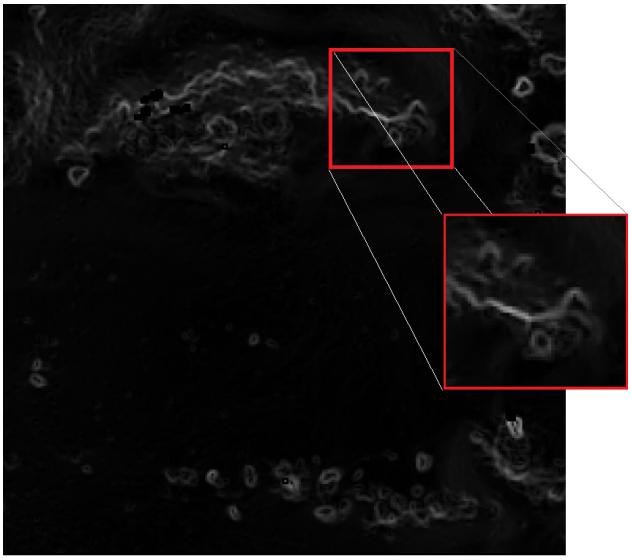


Figure 5: Graphical illustration of how template matching is performed between the local gradient image (template) and the global gradient image (the source).

It is worth mentioning that map-matching, performed with a locally feature-rich terrain that can uniquely be matched to the orbital low resolution map, can potentially reduce the localization error up to the global map resolution, here $\leq 1\,\mathrm{m}$. Figure 6 illustrates the absolute pose correction process.

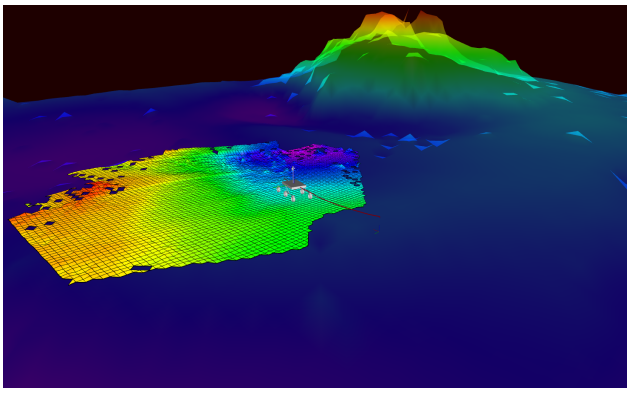


Figure 6: The local (robot-centric) elevation map overlaid on top of a patch of global elevation map. Colors represent the elevation range of values individually for each map.

Since template map matching requires the processing of a large amount of data, a proper execution strategy has to be adopted. Therefore two criteria are evaluated before triggering the pose correction calculation. The first one is to ensure execution efficiency, and checks that a certain threshold distance has been traversed since the last pose correction. This can be balanced depending on the expected quality of the relative localization and the Mission set requirement with respect to permitted absolute drift in localization. The second one ensures execution effectiveness, and checks that the current local map environment is rich in elevation features, by looking at its gradient image, calculating a score by adding all gradient values and comparing it to a certain threshold. This parameter needs to be tuned to the morphological characteristics of the terrain and the size and resolution of the local and global maps.

Overall, the pose correction process could be a relatively time consuming and computationally expensive method. The proper selection of resolution and size of the local and global maps help in reducing the use of resources. Additionally, these two criteria make sure that the process is only activated when needed, i.e.,

long enough distance has been traversed, and when the chances of finding a successful match are high, i.e., the local map roughness is high.

3 Experimental Results

The GA SLAM algorithm explained in section 2 has been implemented as a C++ library [1], integrated, and tested at the Planetary Robotics Lab (PRL, 2018) of the European Space Agency. The conceptual diagram showing the main system architecture components and interfaces to GA SLAM can be seen in Figure 7.

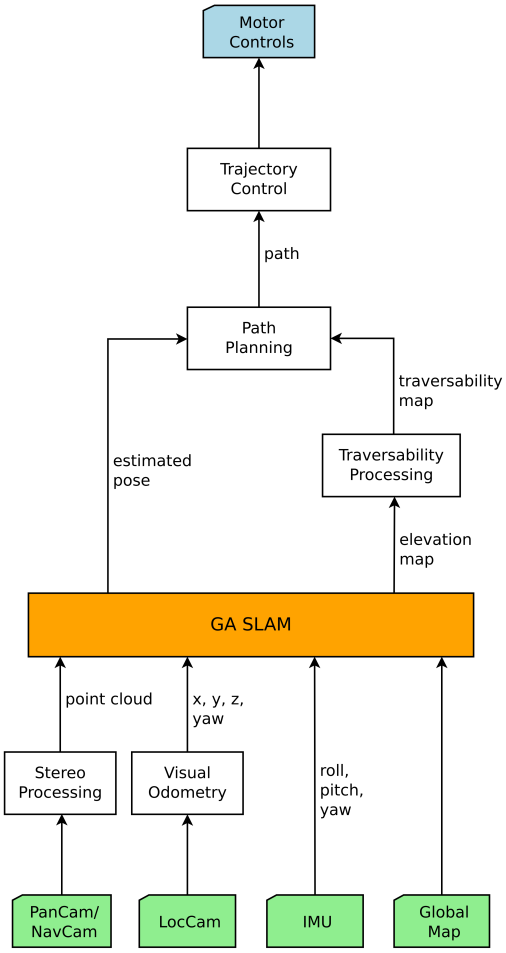


Figure 7: The system architecture conceptual diagram implementation for running operations with onboard autonomous navigation.

The Heavy Duty Planetary Rover (HDPR) lab rover prototype has been used for the experimental validation of the developed algorithm. HDPR (Boukas and Hewitt, 2016; Hewitt et al., 2018), depicted in Figure 9, bears close resemblance, in terms of sensing camera setup, to the ExoMars mission rover (Vago, 2017). Its powerful locomotion system, capable of moving at speeds of up to 1 m/s, makes this rover a very suitable research platform for long range experiments with focus on different sensor integration and data fusion and algorithms for autonomous navigation.

[1] GA SLAM, `https://github.com/ESA-PRL/slam-ga_slam`

3.1 Dataset Collection at the Tenerife Field Test Campaign

In June 2017, the Planetary Robotics Lab performed a field test campaign in Minas de San Jose, at the Teide Volcano National Park in the Spanish island of Tenerife, a highly representative environment of a planetary exploration mission scenario, with a fair distribution of loose and compact soil, slopes and rocks of small, medium and big size (see Figure 8 for reference). The test campaign lasted for 10 days, during which HDPR performed different experiments with the objectives of: sensor dataset collection, teleoperation in dark areas (night-time) and ExoMars-like remote rover operations. While the authors are working on publishing the results and dataset collected during the field test campaign, as previously done for the dataset publication in (Hewitt et al., 2018), part of the collected dataset has been utilized for the experimental validation of the GA SLAM algorithm.

Figure 8: Aerial view of the field test campaign area where the terrain characteristic can be appreciated. The field test area is marked by a dash-lined contour to the right of the road. Left of the road is an area of steep slopes and the direction to the Teide Volcano crater.

In this paper we make use of the dataset collected from different optical camera sensors and for that we explain here the test setup and approach followed for the data acquisition. Geo-referenced traverses to be run by the rover were prepared in advance using Google Earth and verified for safety at the arrival to the field in order to guarantee no obstacles or big slopes would block the path. These traverses were comprised of a set of waypoints that the rover was commanded to follow. For that purpose the rover employed the trajectory control algorithm described in (Filip et al., 2017) and a GNSS receiver for absolute localization, together with Real-Time Kinematic (RTK) corrections coming from the GNSS Base station mounted at the field location. The rover completed a total of 13 km in several separate traverses and gathered sensor data amounting to 3 TB, from the whole set of sensors depicted in Figure 9. For the validation of the GA SLAM algorithm, and in order to follow a standard sensor suite of Mars rover missions, the set of sensor data in Table 1 has been used.

Sensor	Sensor Model	Purpose
LocCam	FLIR Bumblebee 2 (stereo)	Stereo pairs processed at the VO component to produce the odometry input
NavCam	FLIR Bumblebee 3 (stereo)	Point clouds from the 24 cm baseline stereo bench are used as input measurements to the particle filter and data registration modules.
PanCam	2 × FLIR Grasshopper 2	Point clouds from the custom made 50 cm baseline stereo bench are used as input measurements to the particle filter and data registration modules.
IMU	Sensonor Stim300	Augments the pose estimation of the particle filter by adding the roll and pitch measurements and fusing the yaw estimation.
GNSS	Trimble GNSS receiver (BD970) & Trimble Base Station (BX982)	The rover traverse is tracked by the receiver which receives online RTK corrections from the base station to achieve centimetric accuracy. The tracked trajectory is used as ground truth data against which the SLAM pose estimation solution shall be compared.
Orbiter	The SenseFly eBee drone	Maps the field test area and provides the global map with a resolution similar to the maps coming from an orbiter such as HiRISE. Pix4D Mapper Software is used to process the images taken by the drone and produce a 2D orto-mosaic map (Figure 8) and DEMs of the terrain at different resolutions. These DEMs are used at the global pose correction module.

Table 1: Set of sensor data used from the dataset collection for the validation of the GA SLAM algorithm

While the rover did not run the SLAM algorithm online during the traverses in the field, the same software architecture and computer as the one used in the field has been used to execute the experiments explained hereafter. The only difference between the online and offline processes in this case is the fact that image pairs are not coming from the real hardware but from logged binary files. All the processing afterwards is done as if it were running online. Datasets collected in nominal daylight conditions are used when the rover is commanded at a nominal driving speed of 30 cm/s. Given the rover dimensions (Boukas and Hewitt, 2016) and speed, the local map is set to a resolution of 10 cm/cell and a size of

3.2 Experiments on Relative Localization

The purpose of these experiments is to test the accuracy of the Pose Estimation module (see subsection 2.2) and evaluate its capabilities for relative localization. In these experiments the output of the GA SLAM algorithm is compared against the ground truth data collected from the GNSS receiver. Several traverses are run with varying conditions in order to analyse the performance of the SLAM solution in different scenarios. These include a variation of properties of soil, amount of rocks, amount and level of slopes and length. In general, it is understood that feature-rich environments help the visual odometry process and local changes in elevation (such as rocks) can be advantageous for the scan matching process. Note that the weights of the particles in the particle filter are updated according to the fitness score of the scan matching. A total of five traverses are run and their results plotted hereafter. For each traverse a bird's eye view of the trajectory is shown where the ground truth (black), odometry (blue) and SLAM (red) pose paths are overlayed. Each traverse is run three times in order to account for the inherent probabilistic behaviour of the system design and implementation. All the results are summarized and reported in tables together with an average of the three repetitions for each traverse. The bold font numbers in the tables reflect the average error of the SLAM and odometry poses expressed in percentage of the distance traversed at the end of the trajectories.

Trav	Run	Soil	Rocks	Slopes	Length	Acc (Abs) $\Delta\theta$	SLAM Error	Odom. Error
1	1	Hard	Low	Med.	105.5m	310° (3.6°)	0.55m (0.52 %)	2.61m (2.47 %)
1	2	Hard	Low	Med.	105.5m	310° (3.6°)	0.26m (0.25 %)	2.57m (2.44 %)
1	3	Hard	Low	Med.	105.5m	310° (3.6°)	1.01m (0.96 %)	2.56m (2.43 %)
1	Av.	Hard	Low	Med.	105.5m	310° (3.6°)	0.60m (**0.57 %**)	2.58m (**2.45 %**)

Table 2: Summary table of results of Traverse 1

Trav	Run	Soil	Rocks	Slopes	Length	Acc (Abs) $\Delta\theta$	SLAM Error	Odom. Error
2	1	Loose	High	Med.	95m	597° (88°)	0.82m (0.87 %)	2.21m (2.33 %)
2	2	Loose	High	Med.	95m	597° (88°)	1.16m (1.22 %)	2.04m (2.14 %)
2	3	Loose	High	Med.	95m	597° (88°)	1.84m (1.94 %)	2.79m (2.94 %)
2	Av.	Loose	High	Med.	95m	597° (88°)	1.35m (**1.34 %**)	2.48m (**2.46 %**)

Table 3: Summary table of results of Traverse 2

Trav	Run	Soil	Rocks	Slopes	Length	Acc (Abs) $\Delta\theta$	SLAM Error	Odom. Error
3	1	Hard	Low	High	65m	202° (−5.5°)	1.07m (1.65 %)	1.4m (2.15 %)
3	2	Hard	Low	High	65m	202° (−5.5°)	1.52m (2.34 %)	1.74m (2.68 %)
3	3	Hard	Low	High	65m	202° (−5.5°)	0.62m (0.95 %)	0.83m (1.28 %)
3	Av.	Hard	Low	High	65m	202° (−5.5°)	1.07m (**1.65 %**)	1.32m (**2.04 %**)

Table 4: Summary table of results of Traverse 3

Trav	Run	Soil	Rocks	Slopes	Length	Acc (Abs) $\Delta\theta$	SLAM Error	Odom. Error
4	1	Hard	Low	High	107m	385° (−112°)	0.84m (0.79 %)	1.21m (1.13 %)
4	2	Hard	Low	High	107m	385° (−112°)	0.81m (0.76 %)	1.29m (1.21 %)
4	3	Hard	Low	High	107m	385° (−112°)	1.18m (1.10 %)	1.57m (1.47 %)
4	Av.	Hard	Low	High	107m	385° (−112°)	0.94m (**0.88 %**)	1.36m (**1.27 %**)

Table 5: Summary table of results of Traverse 4

Trav	Run	Soil	Rocks	Slopes	Length	Acc (Abs) $\Delta\theta$	SLAM Error	Odom. Error
5	1	Loose	Low	Low	104m	558° (97°)	6.74m (6.48 %)	6.64m (6.38 %)
5	2	Loose	Low	Low	104m	558° (97°)	6.65m (6.39 %)	6.64m (6.38 %)
5	3	Loose	Low	Low	104m	558° (97°)	7.29m (7.52 %)	7.28m (7.51 %)
5	Av.	Loose	Low	Low	104m	558° (97°)	7.06m (**6.81 %**)	7.02m (**6.78 %**)

Table 6: Summary table of results of Traverse 5

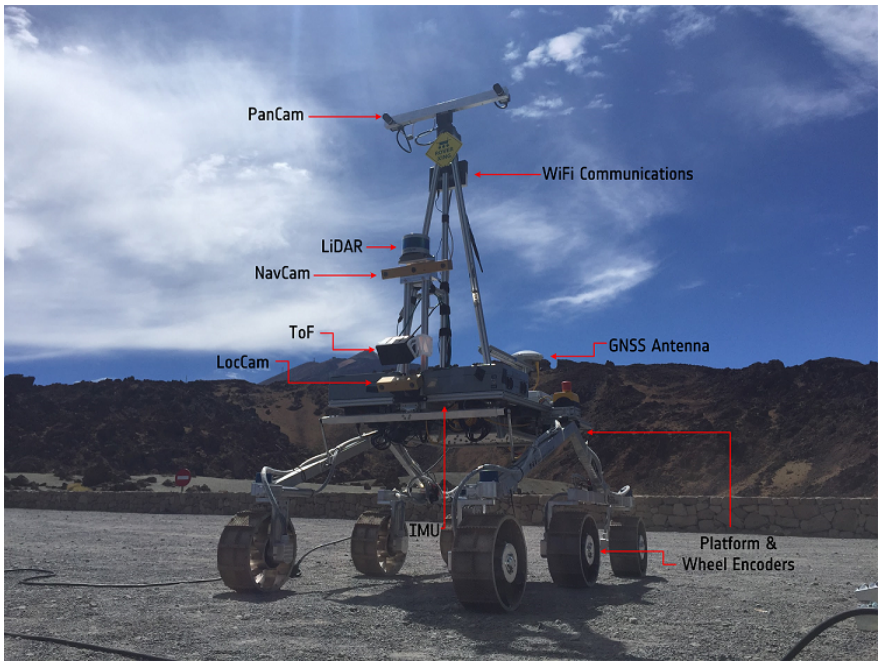
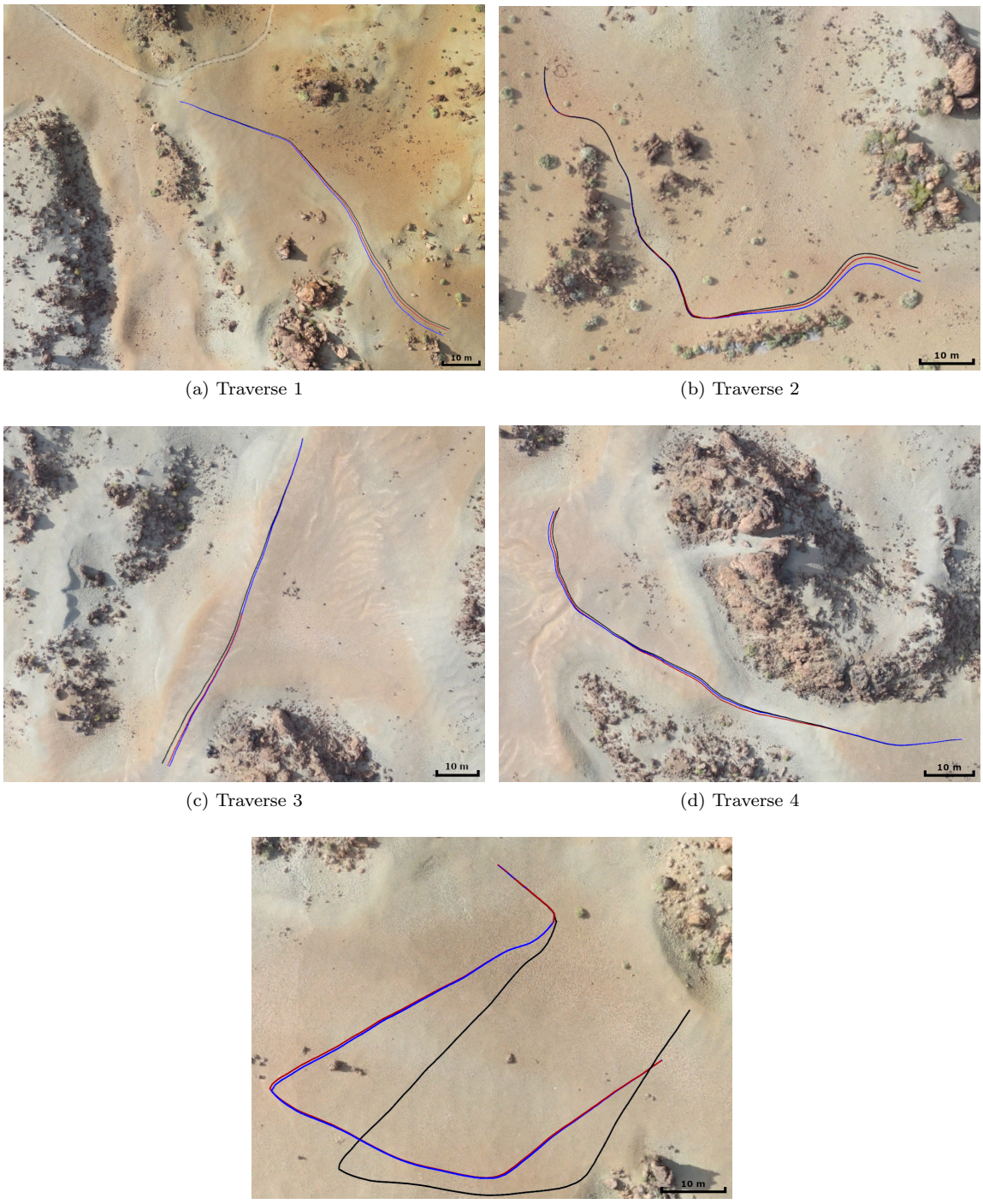


Figure 9: HDPR in Tenerife field test at Teide Volcano.

The traverses in Figure 10 show the overall good performace of the pose estimation module resulting on an improved localization with respect to the odometry input. In some cases the improvement is significant achieving absolute pose estimation errors below 1 %. This result could be compared to the performance of a state-of-the-art VO algorithm, which typically is in the range of 1 to 2 % (Shaw et al., 2013; Kostavelis et al., 2016). Results vary according to the conditions of the traverse. With respect to the soil type we can conclude that, as it was expected, its influence on our approach is not significant. Although this could be a limitation of the dataset used given that the soil was quite homogeneous along the field test site with only some areas having a bit more loose granular soil than the predominant compact or relatively hard ground. Rocks and slopes have a significant positive effect (see Table 3) since these are the elevation features on which the scan matching technique is relying to correct or improve the rover localization. Surprisingly, results on Traverse 1 yield the highest accuracy, i.e. the lowest estimation error (see Table 2), despite the low number of rocks. This can be due to the medium level and amount of slopes and the relatively small accumulated change in orientation (referred to as Acc $\Delta\theta$). Traverses 3 and 4 are very similar in conditions, with considerable uphill and downhill slopes respectively. In both cases the pose estimation is improved with respect to the odometry estimation by 0.39 % (see Table 4 and Table 5). Finally, it is interesting to see the results in Traverse 5 (see Table 6), where the SLAM is run over a traverse with almost no elevation features at all. The significant changes in heading, coupled with the flatness of the terrain made the algorithm incapable of finding the proper corrections. One good thing to remark in this scenario is that the pose estimation module does not add any drift to the odometry estimation and therefore the quality of the result is purely dependant on the quality of the odometry input. An important conclusion after these experiments is then drawn by the fact the SLAM pose estimation can only potentially improve the localization and under certain conditions it improves the estimation considerably.

Finally, different experiments are executed with a varying amount of samples (particles) for the particle filter and varying resampling frequency. This is due to the fact that minimizing computational overhead added by running SLAM is just as important as having an improved relative localization. These results are shown in Figure 11. In particular, we observe that the pose error decreases as we increase the number of particles used in the particle filter (Figure 11a), as it is expected since the continuous space of the problem is modeled more effectively. However, increasing this parameter brings a proportional increase to the computation needs for every iteration of the filter. Regarding the resampling frequency (Figure 11b),



(a) Traverse 1

(b) Traverse 2

(c) Traverse 3

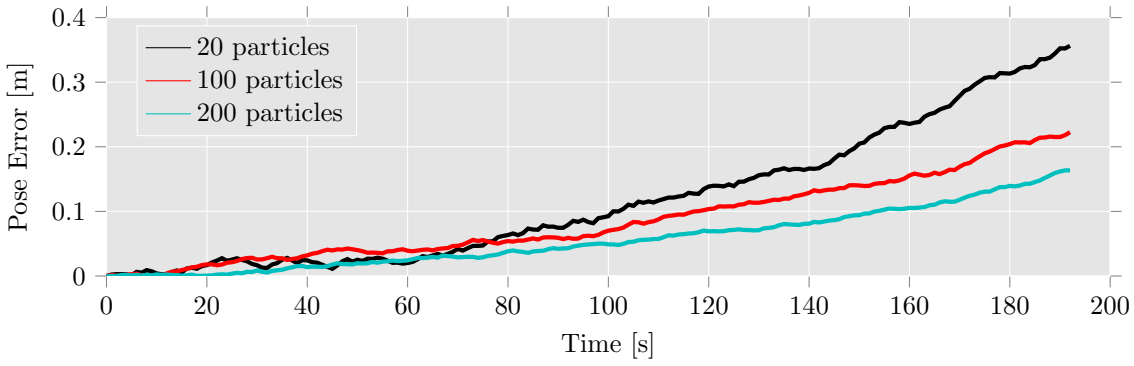
(d) Traverse 4

(e) Traverse 5

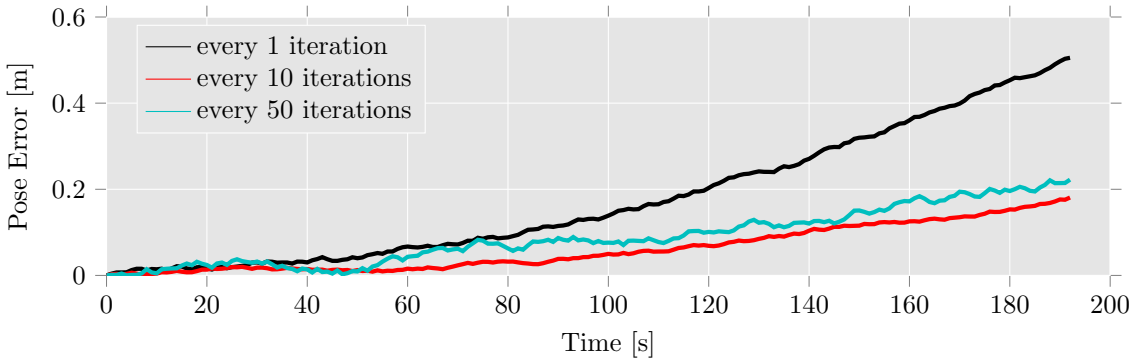
Figure 10: Traverses of experiments on relative localization. Black, blue and red lines represent the ground truth, odometry- and slam-estimated trajectories respectively.

it can be seen that resampling too often or too seldom has an impact on the pose error. This is due to the fact that by resampling, the particle population is gathered near the best belief and as a result other possible solutions are eliminated. Eventually, with an optimal set of configuration parameters the relative localization can reach an accuracy that is below the 1 % of the traversed distance, which is an improvement

with respect to the current state-of-the-art.



(a) Number of particles



(b) Resampling frequency

Figure 11: (a) The influence the number of particles has on the pose error. (b) The influence the resampling frequency parameter has on the error.

3.3 Experiments on Absolute Localization

The objective of this experiment is to validate our proposed solution for global localization and to correct the error in relative localization that accumulates over time. The drift of the pose estimation is an unavoidable problem and it can render the continuation of the navigation unsafe. Therefore, we tested the system in a long-range challenging scenario where localization drifts are manifested and we can assess the applicability of the Pose Correction (see subsection 2.3) module. We evaluate the pose correction functionality in a terrain with low feature richness, i.e., a mainly flat terrain that contains a low amount of elevation features such as rocks. In Figure 12 we can see the results of this experiment showing the terrain in which the test was executed and the traverse inside it. It can be seen that the SLAM estimation (red path) has a notable drift from the ground truth values (black path), similar to the experiment over Traverse 5 shown in the previous section. Close to the end of the traverse, where the presence of elevation features is then more prominent, we can observe the correction of the rover's pose due to a successful match of the local to global elevation map, bringing the pose estimation error down to values smaller than the global map cell size resolution (see Figure 13). The global map used for this experiment had a size of 100×100 m^2, which bounds the area for the template matching process to a reasonable size. As already mentioned in subsection 1.3, the approach used here assumes the rover global location is known within an uncertainty of few tens of meters and therefore is proposed to be used in combination with other techniques for rover global localization. It is also relevant to note that the second criteria used to trigger the pose correction has prevented the process to run during most of the traverse while being on considerably flat areas. Only when the elevation features were rich enough has started the matching process. This important criteria also makes sure that the match

found is certainly unique, unlike in the case of a large flat area, where many flat-to-flat accurate matches could be potentially found, but not trusted.

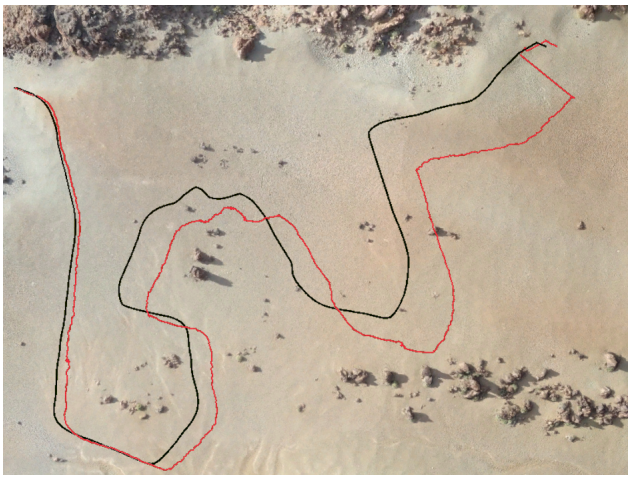


Figure 12: Orthographic views of the environment and the rover's traverses in it. The black and red colors correspond to the ground truth and SLAM paths accordingly. The pose correction is visible in the end of the traverse.

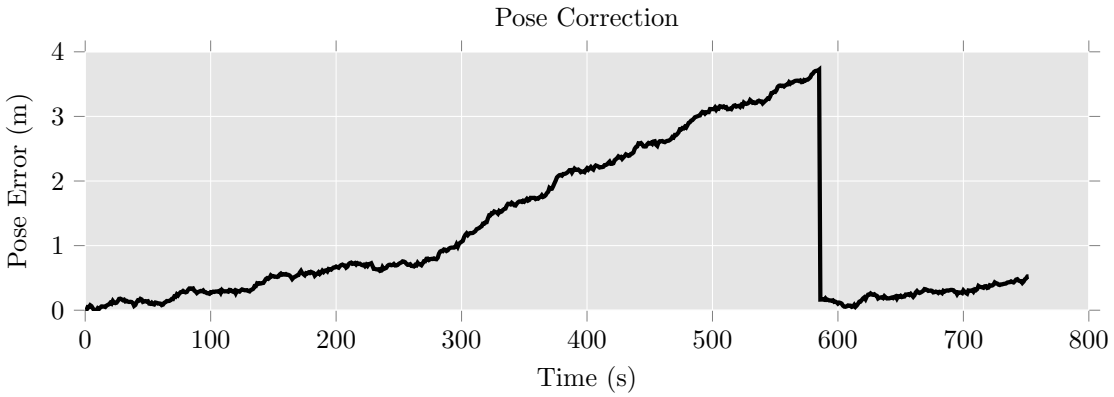


Figure 13: Pose Estimation error over time at absolute localization experiment with local and global map resolutions of 0.1 m/pixel and 0.5 m/pixel respectively. 97.1 % of matching score achieved at correction.

Finally, we want to measure the performance of our global map matching technique for absolute localization with varying combinations of resolutions for both the local and global maps. This is done with the aim to evaluate the suitability of our approach with respect to the available global map data as well as the processing capabilities of the rover. In Table 7, we observe the highest matching score achieved for every combination of resolutions. Note that the resolution of the global map has a greater impact on the map matching score than the resolution of the local map. In fact, a global map with a resolution of 2.0 m/pixel yields poor results for every resolution of the local map while a global map with a resolution of 1.0 m/pixel has the possibility to yield a high score only when the resolution of the local map is high enough. When having global maps of 0.5 m/pixel resolution available, sufficiently high matching scores are achieved for different local map resolutions.

To recap, first, the size and resolution of the local map should be chosen according to the locomotion capabilities of the rover since they impact the limited computational resources of the rover. Therefore, the local map should be detailed and big enough to assess a safe and optimal path to be followed by the rover while trying to minimize these two aspects in order not to overload the onboard processing. Second,

Local map resolution [m]	Global map resolution [m]	Matching score
0.1	0.5	97 %
0.1	1.0	76 %
0.1	2.0	48 %
0.2	0.5	91 %
0.2	1.0	67 %
0.2	2.0	32 %
0.5	0.5	72 %
0.5	1.0	36 %
0.5	2.0	25 %

Table 7: Score of match for combinations of local and global map resolutions.

the resolution of the global map is limited by the available orbiter imagery. As an example, the HiRISE on board camera of the Mars Reconnaissance Orbiter has already produced maps with resolutions of up to 25 cm/pixel over locations of high priority such as potential landing sites for future missions to Mars. We can therefore assume that future Mars exploration missions will have available global maps of sufficient resolution with which our approach can provide absolute corrections for the pose estimation and correct the global drift in localization. Finally, it is important to note that the fact that matching scores of over 90 % have been achieved also demonstrates the good quality of the local map produced and shows its suitability for traversability map computation and further navigation.

4 Conclusion

We described the design of a novel approach for autonomous navigation using SLAM techniques applied to the planetary robotics exploration scenario. Exploiting the conditions in which rovers operate, GA SLAM supports long-range low-supervision autonomous navigation by providing accurate relative localization, a local elevation map from which the traversability for navigation can be easily computed, and a global pose correction method that exploits this same local elevation map and the *a priori* known orbital global map of the planetary terrain. The algorithm has been implemented as a standalone C++ library and integrated in the rover control software architecture of the Planetary Robotics Lab of the European Space Agency. The experimental results of the algorithm using a dataset collected at the Teide Volcano of Tenerife show an improved accuracy in relative localization compared to the visual odometry pose estimation with errors in best cases as low as 0.5 % to 1 % of the traversed distance and the successful correction of the accumulated drift of the estimated pose within the absolute global map, with a correction precision equivalent to the resolution of the orbital map.

The aim of this research is the design, implementation and in test validation of an autonomous navigation approach for planetary rovers. The SLAM approach presented in this paper is an step in that direction, fulfilling two core necessary modules for autonomous exploration, i.e. self-localization and terrain-mapping. The design drivers that have led this work come given from the application use case and, as we already mentioned, SLAM approaches on planetary rovers have the relevant constraint of the limited on board processing power. Design choices have tried to balance the efficiency of the system and that took the final implementation to show its limitations in flat terrains. To overcome this limitations would require adding and tracking other kind of features within the SLAM module, increasing consequently its computational load. Instead we have shown the applicability of a global localization module that efficiently reuses the information that already exists in the SLAM and serve to correct the drift that inevitably increases over distance. Yet, the challenge of making all these processes computationally even more efficient remains, given the aforementioned scarce resource of rover avionics. With that in mind, a possible further extension of the proposed particle filter algorithm would be to model the space of the problem, i.e. localization, in a discretized way instead of a continuous one. This can be achieved by sampling particles in a 2D grid that

has the same resolution as the local map. As a result, particles that fall in the same cell of the grid are eliminated and are replaced by new particles. Another improvement would be to exploit the uncertainty of the robot's pose to improve the quality of the local map. This can be accomplished using a neighborhood fusion method (Fankhauser et al., 2014) by fusing nearby cells for each cell of the map according to the variance of the robot's translation. The variance values could be extracted directly from the particle filter, by calculating the distribution of the particles, or by using the motion model of the robot.

Acknowledgement

The work presented in this paper was supported by the Automation & Robotics Section of ESA in collaboration with the Aristotle University of Thessaloniki.

References

Azkarate, M., Kapellos, K., Hewitt, R. A., Boukas, E., Joudrier, L., Poulakis, P., Bussi, D., Ferentino, E., and Visentin, G. (2016). Remote Rover Operations: Testing the ExoMars Egress Case. In 13th International Symposium on Artificial Intelligence, Robotics, and Automation in Space (iSAIRAS).

Bampis, L., Amanatiadis, A., and Gasteratos, A. (2018). Fast loop-closure detection using visual-word-vectors from image sequences. The International Journal of Robotics Research, 37(1):62–82.

Barfoot, T. D. (2005). Online visual motion estimation using fastslam with sift features. In 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 579–585. IEEE.

Boukas, E., Gasteratos, A., and Visentin, G. (2018). Introducing a globally consistent orbital-based localization system. Journal of Field Robotics, 35(2):275–298.

Boukas, E. and Hewitt, R. (2016). HDPR: A mobile testbed for current and future rover technologies. In International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS).

Bousquet, P. (2011). The cnes contribution to in-situ exploration: robotic implications. In 11th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA).

Brenna, M. (2008). Scan matching, covariance estimation and SLAM: models and solutions for the scanSLAM algorithm. PhD thesis, Politecnico di Milano.

Carle, P. J., Furgale, P. T., and Barfoot, T. D. (2010). Long-range rover localization by matching lidar scans to orbital elevation maps. Journal of Field Robotics, 27(3):344–370.

Cheng, Y., Maimone, M., and Matthies, L. (2005). Visual odometry on the mars exploration rovers. In 2005 IEEE International Conference on Systems, Man and Cybernetics, volume 1, pages 903–910. IEEE.

Chiodini, S., Pertile, M., Debei, S., Bramante, L., Ferrentino, E., Villa, A. G., Musso, I., and Barrera, M. (2017). Mars rovers localization by matching local horizon to surface digital elevation models. In 2017 IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace), pages 374–379. IEEE.

Clerc, S., Martella, P., Durrant, D., Bertsch, N., and Dussy, S. (2009). Development of the european imu for space applications. In AIAA Guidance, Navigation, and Control Conference, page 5870.

Correal, R. and Pajares, G. (2011). Onboard autonomous navigation architecture for a planetary surface exploration rover and functional validation using open source tools. In 11th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA).

Cozman, F. and Krotkov, E. (1998). Automatic mountain detection and pose estimation for teleoperation of lunar rovers. In Experimental Robotics V, pages 207–215. Springer.

Cozman, F., Krotkov, E., and Guestrin, C. (2000). Outdoor visual position estimation for planetary rovers. Autonomous Robots, 9.

Cremean, L. B. and Murray, R. M. (2005). Uncertainty-based sensor fusion of range data for real-time digital elevation mapping (rtdem). In IEEE International Conference on Robotics and Automation (ICRA), pages 18–22.

Douc, R. and Cappé, O. (2005). Comparison of resampling schemes for particle filtering. In 4th International Symposium on Image and Signal Processing and Analysis, pages 64–69. IEEE.

Doucet, A., De Freitas, N., and Gordon, N. (2001). An introduction to sequential monte carlo methods. In Sequential Monte Carlo methods in practice, pages 3–14. Springer.

Doucet, A., De Freitas, N., Murphy, K., and Russell, S. (2000). Rao-blackwellised particle filtering for dynamic bayesian networks. In Sixteenth conference on Uncertainty in artificial intelligence.

Durrant, D., Utton, M., Whitley, E., and Kowaltschek, S. (2017). Evolution of Next Generation European MEMS Inertial Products. In 10th International ESA Conference on Guidance, Navigation and Control Systems.

Fankhauser, P., Bloesch, M., Gehring, C., Hutter, M., and Siegwart, R. (2014). Robot-centric elevation mapping with uncertainty estimates. In Mobile Service Robotics, pages 433–440. World Scientific.

Filip, J., Azkarate, M., and Visentin, G. (2017). Trajectory control for autonomous planetary rovers. In 14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA).

Grisetti, G., Kummerle, R., Stachniss, C., and Burgard, W. (2010). A tutorial on graph-based slam. IEEE Intelligent Transportation Systems Magazine, 2(4):31–43.

Grisettiyz, G., Stachniss, C., and Burgard, W. (2005). Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In IEEE international conference on robotics and automation (ICRA), pages 2432–2437. IEEE.

Hahnel, D., Fox, D., Burgard, W., and Thrun, S. (2003). A highly efficient fastslam algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In Proceedings of Conference Intelligent Robots and Systems (IROS).

Hashemi, N. S., Aghdam, R. B., Ghiasi, A. S. B., and Parastoo, F. (2016). Template matching advances and application in image analysis. American Scientific Research Journal for Engineering, Technology, and Sciences.

Hewitt, R. A., Boukas, E., Azkarate, M., Pagnamenta, M., Marshall, J. A., Gasteratos, A., and Visentin, G. (2018). The katwijk beach planetary rover dataset. International Journal Robotics Research (IJRR), 37(1):3–12.

Hidalgo-Carrió, J., Hennes, D., Schwendner, J., and Kirchner, F. (2017). Gaussian process estimation of odometry errors for localization and mapping. In 2017 IEEE International Conference on Robotics and Automation (ICRA), pages 5696–5701. IEEE.

Hidalgo-Carrió, J., Poulakis, P., and Kirchner, F. (2018). Adaptive localization and mapping with application to planetary rovers. Journal of Field Robotics, 35(6):961–987.

Hinks, T., Carr, H., Truong-Hong, L., and Laefer, D. F. (2013). Point cloud data conversion into solid models via point-based voxelization. Journal of Surveying Engineering, 139(2):72–83.

Huesing, J., Rebuffat, D., Falkner, P., Vago, J., and Pickering, A. (2014). Cdf study report: Marsfast. assessment of an esa fast mobility mars rover rover.

Hwangbo, J. W., Di, K., and Li, R. (2009). Integration of orbital and ground image networks for the automation of rover localization. In ASPRS 2009 Annual Conference.

Jayanthi, N. and Indu, S. (2016). Comparison of image matching techniques. International Journal of Latest Trends in Engineering and Technology, 7(3):396–401.

Jefferies, M. E. and Yeap, W.-K. (2008). Robotics and cognitive approaches to spatial mapping. Springer Tracts in Advanced Robotics.

Jet Propulsion Lab (2018a). Mars Exploration Rovers. Available at: `http://marsrovers.jpl.nasa.gov/home/index.html`. Last accessed on August 2018.

Jet Propulsion Lab (2018b). Mars Science Laboratory. Available at: `https://mars.nasa.gov/msl/`. Last accessed on August 2018.

Kostavelis, I., Boukas, E., Nalpantidis, L., and Gasteratos, A. (2016). Stereo-based visual odometry for autonomous robot navigation. International Journal of Advanced Robotic Systems, 13(1):21.

Kostavelis, I., Boukas, E., Nalpantidis, L., Gasteratos, A., and Rodrigalvarez, M. A. (2011). Spartan system: Towards a low-cost and high-performance vision architecture for space exploratory rovers. In IEEE International Conference on Computer Vision Workshops (ICCV Workshops), pages 1994–2001. IEEE.

Kostavelis, I., Nalpantidis, L., Boukas, E., Rodrigalvarez, M. A., Stamoulias, I., Lentaris, G., Diamantopoulos, D., Siozios, K., Soudris, D., and Gasteratos, A. (2014). Spartan: Developing a vision system for future autonomous space exploration robots. Journal of Field Robotics, 31(1):107–140.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). G2o: A general framework for graph optimization. In 2011 IEEE International Conference on Robotics and Automation, pages 3607–3613. IEEE.

Lentaris, G., Stamoulias, I., Diamantopoulos, D., Maragos, K., Siozios, K., Soudris, D., Rodrigalvarez, M., Lourakis, M., Zabulis, X., Kostavelis, I., Nalpantidis, L., Boukas, E., and Gasteratos, A. (2015). Spartan/sextant/compass: Advancing space rover vision via reconfigurable platforms. In Applied Reconfigurable Computing, volume 9040 of Lecture Notes in Computer Science, pages 475–486, Germany. Springer.

Li, R., Di, K., Howard, A. B., Matthies, L., Wang, J., and Agarwal, S. (2007). Rock modeling and matching for autonomous long-range mars rover localization. Journal of Field Robotics, 24(3):187–203.

Maimone, M. (2017). The evolution of autonomous capabilities on nasas mars rovers. In Southern California Robotics Symposium. https://www.youtube.com/watch?v=u4-4x8GhE6Y.

Maimone, M., Cheng, Y., and Matthies, L. (2007). Two years of visual odometry on the mars exploration rovers. Journal of Field Robotics, 24(3):169–186.

Matthies, L., Maimone, M., Johnson, A., Cheng, Y., Willson, R., Villalpando, C., Goldberg, S., Huertas, A., Stein, A., and Angelova, A. (2007). Computer vision on mars. International Journal of Computer Vision, 75(1):67–92.

Merlo, A., Larranaga, J., and Falkner, P. (2013). Sample fetching rover (sfr) for msr. In 12th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA).

Montemerlo, M. and Thrun, S. (2003). Simultaneous localization and mapping with unknown data association using fastslam. In IEEE International Conference on Robotics and Automation (ICRA), volume 2, pages 1985–1991. IEEE.

Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. (2002). Fastslam: A factored solution to the simultaneous localization and mapping problem. In Eighteenth National Conference on Artificial Intelligence.

Moreno, S. (2013). Cnes robotics activities: towards long distance ob decision-making navigation. In 12th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA).

Mur-Artal, R. and Tardós, J. D. (2017). ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. IEEE Transactions on Robotics, 33(5):1255–1262.

Murphy, K. and Russell, S. (2001). Rao-blackwellised particle filtering for dynamic bayesian networks. In Sequential Monte Carlo methods in practice, pages 499–515. Springer.

Murphy, K. P. (1999). Bayesian map learning in dynamic environments. In Proceedings of the 12th International Conference on Neural Information Processing Systems.

Ochoa, B. and Belongie, S. (2006). Covariance propagation for guided matching. In Workshop on Statistical Methods in Multi-Image and Video Processing (SMVP), volume 83.

PRL (2018). Planetary Robotics Lab. Available at: `https://www.esa.int/Our_Activities/Space_Engineering_Technology/Automation_and_Robotics/About_the_Planetary_Robotics_Lab`. Last accessed on August, 2018.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In International Conference on Computer Vision (ICCV), pages 2564–2571. IEEE.

Rusinkiewicz, S. and Levoy, M. (2001). Efficient variants of the icp algorithm. In Third International Conference on 3-D Digital Imaging and Modeling, pages 145–152. IEEE.

Rusu, R. B. and Cousins, S. (2011). 3d is here: Point cloud library (pcl). In IEEE International Conference on Robotics and Automation (ICRA), pages 1–4. IEEE.

Shaw, A., Woods, M., Churchill, W., and Newman, P. (2013). Robust visual odometry for space exploration. In 12th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA).

Sim, R., Griffin, M., Shyr, A., and Little, J. J. (2005). Scalable real-time vision-based slam for planetary rovers. In IEEE IROS Workshop on Robot Vision for Space Applications, pages 16–21.

Siozios, K., Diamantopoulos, D., Kostavelis, I., Boukas, E., Nalpantidis, L., Soudris, D., Gasteratos, A., Avilés, M., and Anagnostopoulos, I. (2011). SPARTAN project: Efficient implementation of computer vision algorithms onto reconfigurable platform targeting to space applications, pages 1–9. 6th International Workshop on Reconfigurable Communication-Centric Systems-on-Chip, ReCoSoC 2011 - Proceedings.

Stein, F. and Medioni, G. (1992). Structural indexing: Efficient 3-d object recognition. Pattern Analysis and Machine Intelligence.

Thrun, S., Burgard, W., and Fox, D. (2005). Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press.

Thrun, S. et al. (2006). Stanley: The robot that won the darpa grand challenge. Journal of Field Robotics.

Thrun, S., Liu, Y., Koller, D., Ng, A., Ghahramani, Z., and Durrant-Whyte, H. (2004). Simultaneous localization and mapping with sparse extended information filters. International Journal Robotics Research (IJRR), 23(7-8):693–716.

Tong, C. H., Barfoot, T. D., and Dupuis, E. (2011). 3d slam for planetary worksite mapping. In Proceedings of Conference Intelligent Robots and Systems (IROS).

Tong, C. H., Barfoot, T. D., and Dupuis, E. (2012). Three-dimensional slam for mapping planetary work site environments. Journal of Field Robotics.

Vago, Jorge L., e. a. (2017). Habitability on Early Mars and the Search for Biosignatures with the ExoMars Rover. Astrobiology, 17(6-7). Special Collection of Papers: ExoMars Rover Mission.

Wilson, M. (2013). Mars 2020 Mission Concept. Available at: `https://www.nasa.gov/sites/default/files/files/3_Mars_2020_Mission_Concept.pdf`. Last accessed on August 2018.

Woods, M., Shaw, A., Tidey, E., Pham, B. V., Lacroix, S., Mukherji, R., Maddison, B., Cross, G., Kisdi, A., Tubby, W., Visentin, G., and Chong, G. (2014). Seeker—autonomous long-range rover navigation for remote exploration. Journal of Field Robotics.

Zhang, Z. (1994). Iterative point matching for registration of free-form curves and surfaces. In International Journal of Computer Vision.