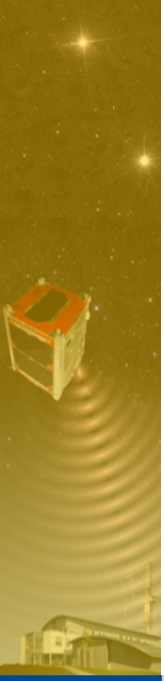


Institut für Informatik
Lehrstuhl für Robotik
Prof. Dr. A. Nüchter
Prof. Dr. K. Schilling

Forschungsberichte
in der Robotik

Research Notes
in Robotics

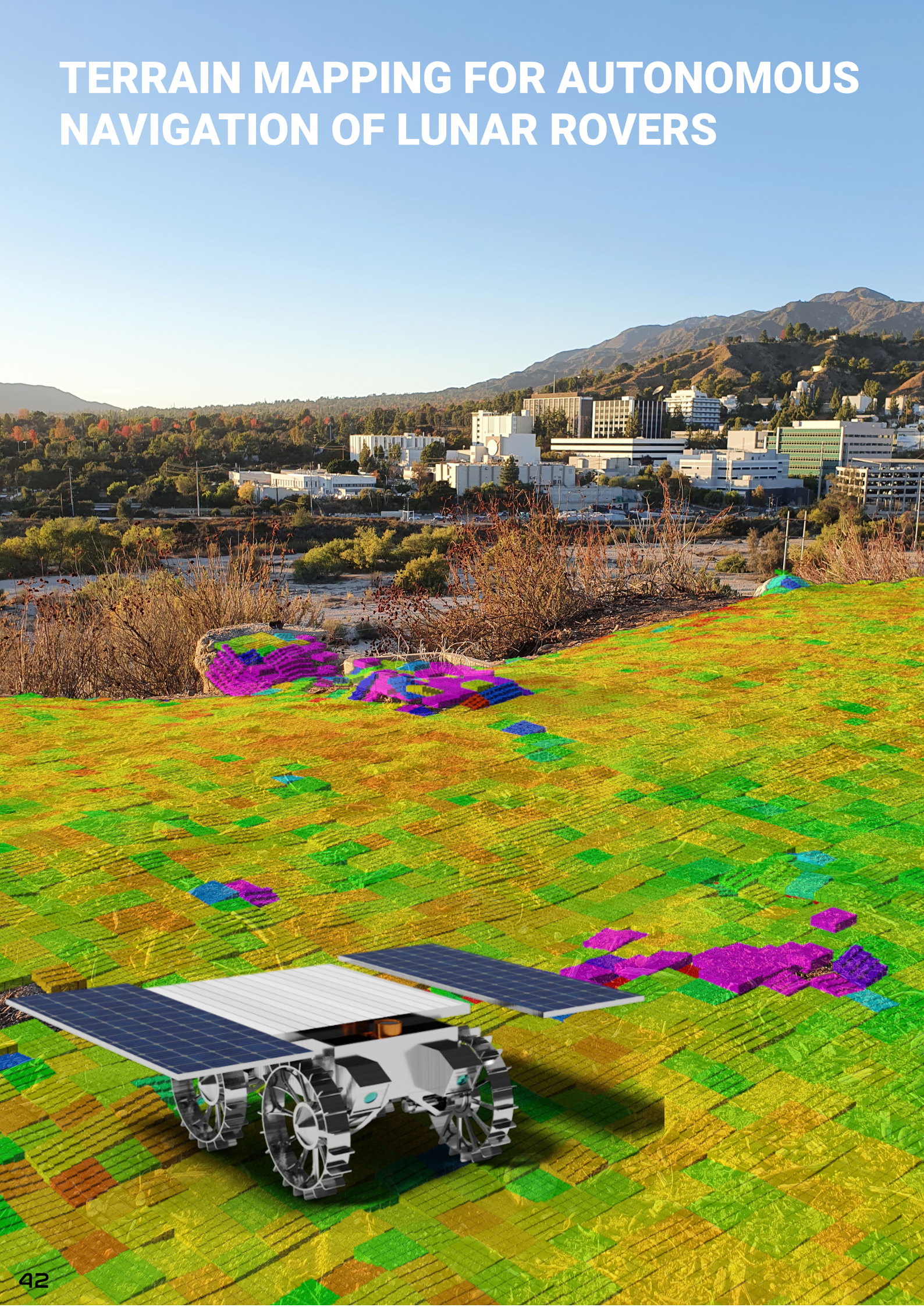
Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

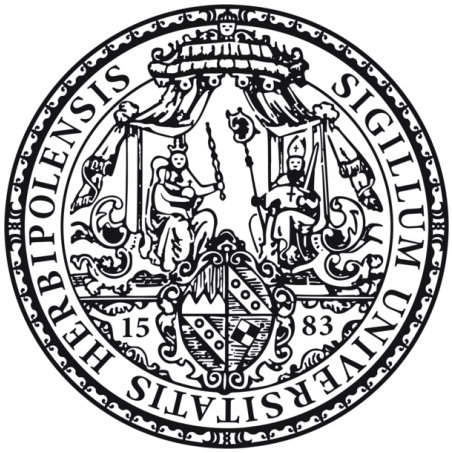
Band 29

Lennart Werner

Terrain Mapping for Autonomous Navigation of Lunar Rovers

TERRAIN MAPPING FOR AUTONOMOUS NAVIGATION OF LUNAR ROVERS





JULIUS MAXIMILIANS UNIVERSITÄT
WÜRZBURG CHAIR OF COMPUTER SCIENCE VII
ROBOTICS AND TELEMATICS

Thesis

TERRAIN MAPPING FOR AUTONOMOUS NAVIGATION OF LUNAR ROVERS

A thesis submitted to attain the degree of

MASTER OF SCIENCE of JULIUS MAXIMILIANS UNIVERSITY OF WÜRZBURG
(M.Sc.)

presented by

LENNART WERNER
B.Sc.

born on 11 August 1998
Lübeck, Germany

Prof. Dr. A. Nüchter, examiner
Prof. Dr. R. Brockers, advisor

December 2022

Lennart Werner: *Terrain Mapping for Autonomous Navigation of Lunar Rovers*

ABSTRACT

Autonomous mobile robots operating in unknown terrain have to guide their drive decisions through local perception. Local mapping and traversability analysis is essential for safe rover operation and low level locomotion. This thesis deals with the challenge of building a local, robot centric map from ultra short baseline stereo imagery for height and traversability estimation.

Several grid-based, incremental mapping algorithms are compared and evaluated in a multi size, multi resolution framework. A new, covariance based mapping update is introduced, which is capable of detecting sub-cellsize obstacles and abstracts the terrain of one cell as a first order surface.

The presented mapping setup is capable of producing reliable terrain and traversability estimates under the conditions expected for the Cooperative Autonomous Distributed Robotic Exploreration (CADRE) mission.

Algorithmic- and software architecture design targets high reliability and efficiency for meeting the tight constraints implied by CADRE's small on-board embedded CPU.

Extensive evaluations are conducted to find possible edge-case scenarios in the operating envelope of the map and to confirm performance parameters. The research in this thesis targets the CADRE mission, but is applicable to any form of mobile robotics which require height- and traversability mapping.

ZUSAMMENFASSUNG

Autonome mobile Roboter, die in unkartiertem Terrain operieren, müssen ihre Fahrentscheidungen durch lokale Wahrnehmung steuern. Lokale Kartierung und Passierbarkeitsanalysen sind der Schlüssel für einen sicheren Betrieb des Roboters und die Fortbewegung. Diese Arbeit beschäftigt sich mit der Herausforderung, eine lokale, roboterzentrierte Karte für Höhen- und Passierbarkeitsanalysen aus Stereobildern zu erstellen.

Mehrere inkrementelle Kartierungsalgorithmen werden verglichen und in einem Framework mit verschiedenen Layern für Größen und Auflösungen implementiert und verglichen. Ein neues, kovarianzbasiertes Kartierungsupdate wird eingeführt, das in der Lage ist, Hindernisse unterhalb der Zellgröße zu erkennen. Dieser Algorithmus abstrahiert die Umgebung einer Zelle als Oberfläche erster Ordnung.

Das vorgestellte Kartierungssystem ist in der Lage, zuverlässige Gelände- und Durchquerbarkeitsschätzungen unter den CADRE Bedingungen zu liefern.

Das Design der Algorithmen- und Software-Architektur zielt auf hohe Zuverlässigkeit und Effizienz ab, um die engen Vorgaben der eingebetteten CPUs an Bord zu wahren.

Umfassende Evaluierungen werden durchgeführt, um mögliche Grenzszenarien im Betriebsbereich der Karte zu finden und die Leistungsparameter zu bestätigen. Die Forschung in dieser Arbeit zielt auf die CADRE-Mission ab, ist aber auf jede Form der mobilen Robotik anwendbar, die Höhen- und Durchquerbarkeitsschätzungen erfordert.

Space is big.
You just won't believe how vastly, hugely, mind-bogglingly big it is.
I mean, you may think it's a long way down the road to the chemist's, but
that's just peanuts to space.

— The Hitchhiker's Guide to the Galaxy, Douglas Adams

ACKNOWLEDGMENTS

The research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the JPL Visiting Student Research Program (JVSRP) and the National Aeronautics and Space Administration.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

CADRE Cooperative Autonomous Distributed Robotic Exploreration
CALTECH California Institute of Technology
CCM Converging Covariance Map
CLPS Commercial Lunar Payload Services
DEM Digital Elevation Model
FoV Field of View
GNC Guidance Navigation and Control
IMU Inertial Measurement Unit
ICP Iterative Closest Points
JDEM JPL Digital Elevation Model
JPL Jet Propulsion Laboratory
LIDAR Light Detection And Ranging
MSH Mars Science Helicopter
OMG Optimal Mixture of Gaussians
pose position and orientation
ROS Robot Operating System
SRH Sample Return Helicopter
tof time of flight
UWB Ultra Wide Band
VIO Visual Inertial Odometry
VO Visual Odometry

1 INTRODUCTION

This chapter provides an introduction to the work described within this thesis. Besides the motivation behind the presented work, the associated Lunar mission is described briefly and the used notation is defined.

1.1 MOTIVATION

Long signal runtimes, limited bandwidth and the ever-increasing complexity of modern space missions require the advancement of non-interactive driving and operation modes [1]. Already today, flagship planetary exploration missions such as Mars 2020 greatly benefit from basic autonomy modes [2, 3]. They enable for example *beyond sensor coverage driving* or the on-board selection of potentially relevant destinations and on-board scheduling.

While single rover missions are proven tools for scientific success, the associated value of such systems does not leave room for potentially higher risk and reward operations. Rapid advancements in miniaturization of electronics and computational power are enabling the use of distributed (swarm) robotic systems for near future planetary exploration [4]. The advanced mission profile for such systems envisions only high-level operator interaction while low-level tasks, such as orbit planning and driving, are performed autonomously. NASA is planning to launch a technology demonstration mission demonstrating autonomous cooperative driving on the Moon in 2024 [5].

An essential part of higher autonomy operations is the perception and understanding of the terrain surrounding the rover. A reliable terrain model of traversable and non-traversable areas is essential for safe operation of the rovers. Since orbital data of the moon is not available at a high enough resolution to detect drive hazards globally, maps need to be created with on-board perception. Even though stereo imagery based obstacle avoidance is an easy task for most humans, transferring this skill to robotic systems is part of ongoing research. Using cameras for depth estimation is a widely applied technique for obstacle avoidance due to the general robustness, lack of moving parts, affordability, low power consumption and adjustable range of operation. This makes stereo depth the obvious choice for lunar traversability mapping.

1.2 JET PROPULSION LABORATORY

World's leading institution for robotic space exploration, the Jet Propulsion Laboratory (JPL) is located in Pasadena, California. Federally funded by NASA and managed by California Institute of Technology (CALTECH), JPL is home to about 6300 employees.

Which started as a group of Caltech graduate students and amateur rocket enthusiasts around professor Theodore von Kármán in the 1930s [1] quickly became one of the hot spots for space exploration. Originally solely focussed on rocketry and later part of the military, NASA took over the laboratory in 1958 but re-purposed it for space probes and planetary research.

Ever since, JPL has sent probes to all of our solar systems planets and is currently managing 20 spacecraft and eight major instruments conducting active missions. [2] Mighty things are dared.

1.3 CADRE

The work in this thesis is conducted to support the CADRE mission. CADRE's technical goal is to demonstrate cooperative autonomous driving on the moon. Being part of NASA's Space Technology Office's Game Changing Development Program [6] commercially available components are used to design and develop three shoebox-sized rovers and perform autonomous mission operation and formation driving on the lunar surface. Figure 1.1 shows an early test bed version of the CADRE rovers.

The CADRE mission itself is part of the Commercial Lunar Payload Services (CLPS) [7] initiative and will be delivered to the moon by a commercial lander from *Intuitive Machines*. As a landing site the *Reiner Gamma* magnetic anomaly [8] has been selected. The small magnetosphere around the landing area has interfered with solar winds and prevented the area from the moon typical darkening of the regolith. This special landing area at 7.585° N, 58.725° W therefore renders a scientifically interesting target. The scientific payload of CADRE is a multi device ground penetrating radar. Radar operation greatly benefits from the flexible placements of agents to change the antenna baseline during operation. Additionally, stereo cameras will be used to map the terrain. The landing area is marked in Figure 1.2.

Currently, the targeted landing date is January 1 to end of June 2024.

1 https://www.jpl.nasa.gov/who-we-are/history
2 https://www.jpl.nasa.gov/who-we-are

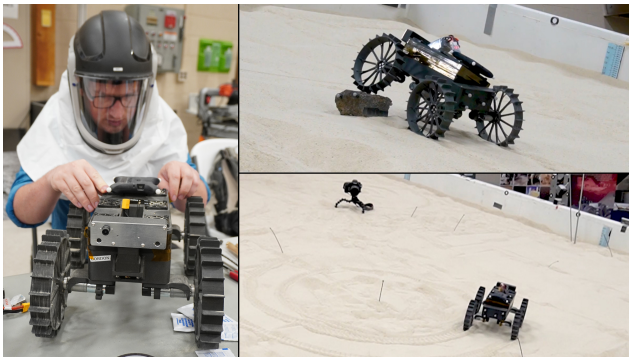


Figure 1.1: CADRE Mobility Test Bed On Regolith Simulant.
Source : [9]

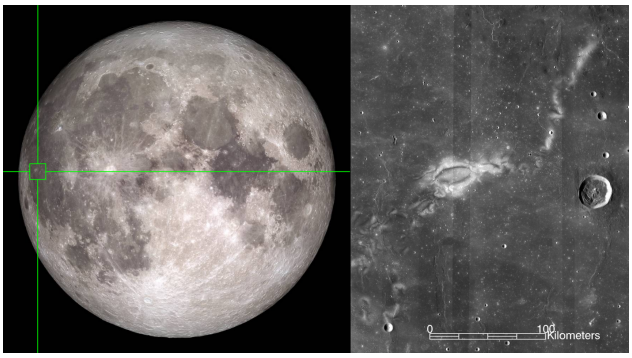


Figure 1.2: Position of the Reiner Gamma Anomaly.
Source : [10, 11]

1.4 PROBLEM FORMULATION ROVER MAPPING

Hazard detection and avoidance is one of the most integral parts of each rover mission. The rovers' capability of distinguishing traversable from non-traversable terrain heavily relies on a solid understanding of the surrounding area.

Maintaining a usable traversability estimation for the rover requires the completion of three main tasks:

- Perception of the environment and of the robots' position and orientation (pose).
- Temporal fusion of the measurements to form a computationally usable representation of the area.
- Qualitative evaluation of the map for traversability analysis.

The CADRE mission relies on dense depth reconstruction from stereo for sensing the environment. This technique is well researched and widely applied [12]. Two cameras with known and fixed distance (baseline) observe the same scene. By finding the pixel disparity for each element of the image a depth map can be created.

As a requirement for using this depth image in mapping the pose of the camera with respect to the environment needs to be known. Typically, this is achieved by combining Visual Odometry (VO) Methods with Inertial measurement systems. VO uses a time series of images to estimate the ego motion of the capturing system by only considering data from one or more imaging sensors [13, 14]. Adding Inertial measurements to the estimation provides increased update rate, precision and robustness [15]. Those Systems are called Visual Inertial Odometry (VIO) and are commonly used for robotic mapping.

Since both, the depth measurements and pose estimates are noisy, they have to be fused and post-processed to get a reliable map. Depth measurements are typically interpreted as a point cloud and transformed into the environment-fixed coordinate frame.

To resolve temporally occluded areas, discard outliers and cope with the stereo error dependent noise, measurements are fused into a model of the environment. Since stereo error (measurement uncertainty) increases with distance, multiple map resolutions have to be available to fuse the measurement at a resolution which matches the measurement precision in order to avoid aliasing effects. This implies a multi resolution / multi size map centered around the robot.

To provide useful information for the path planner, the existing map needs to be processed for segmentation of traversable and non-traversable

areas. Non-traversable areas could be implied by the presence of large rocks, high inclinations or craters.

Furthermore, the cooperative nature of CADRE requires an easy way to fuse the robot-centric locally generated maps together to form a global terrain representation on team level.

A main aspect of the presented challenge is the feasibility of the algorithms for space applications. Tight runtime and memory constraints are present in current space qualified computing units. The general necessity for explainability and simplicity is imposed by the rigorous qualification process of flight software. All implemented techniques must be thoroughly evaluated and tested.

1.5 SCOPE OF THIS THESIS

First and foremost, this thesis communicates the scientific contributions from the work on mapping for CADRE. Furthermore, it serves as a reference to retrace the decisions which lead to the presented conclusions. As a technical report the used methodology is described in detail to support potential reconstruction of the results.

The main scientific contribution from this work is the development of the CCM algorithm.

An extensive literature review of existing mapping techniques has been performed to base this work on the current state of the art. As a design space survey a variety of mapping techniques are evaluated in basic simulation to form candidates for the CADRE mapping.

A tailored multi size multi resolution mapping framework has been developed and deployed in preparation for flight software.

The mapping candidates are evaluated on a high fidelity simulation and on the rover test bed. Their features and drawbacks are discussed, and a final decision is made. Concluding, findings are summarized, and the final setup is presented.

1.6 THESIS OUTLINE

The chapters of this thesis have the following content:

- **Chapter 2** covers fundamental concepts needed for further understanding of the thesis' content.
- **Chapter 3** discusses the current state of the art and presents existing mapping solutions with their advantages and drawbacks.

- **Chapter 4** presents the multi size multi resolution map used for the CADRE mission. This chapter explains the software concept and architecture.
- **Chapter 5** describes the CCM update algorithm which is used in the final flight map.
- **Chapter 6** compares the for CADRE most promising mapping techniques with a preliminary simulation setup.
- **Chapter 7** evaluates and compares the main mapping techniques on a high fidelity simulator and on a real rover tested.
- **Chapter 8** concludes the thesis by discussion the final mapping choice for the flight hardware and suggestions for future research.

1.7 NOTATION

Due to the variety in notation styles used in literature, this section defines the notation within this thesis. Especially stochastic elements require definition. Table 1.1 specifies the used notation.

Stochastic Notation		
Element	Notation	Description
Mean[x]	μ_x or $\bar{x}$	Mean in x- Direction
Std[x]	σ_x or σ_{xx}	Stdandard Deviation in x
Var[x]	σ_x^2 or σ_{xx}^2	Variance in x
Cov[x, y]	σ_{xy}^2	Covariance xy

Table 1.1: Stochastic Notation

The ambiguity σ_x^2 or σ_{xx}^2 is introduced to support a more convenient matrix form. A full 3 × 3 Covariance matrix thus looks like Equation (1.1).

$$Cov_{3\times3} = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{xz}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 & \sigma_{yy}^2 \\ \sigma_{zx}^2 & \sigma_{yy}^2 & \sigma_{zz}^2 \end{bmatrix} \tag{1.1}$$

The Standard Deviation ambiguity serves consistency.

2

PRELIMINARIES

This chapter explains the basic concepts required for the topics covered in this thesis. Additional to covering the depth from stereo technique, a short primer about variances is included.

2.1 INTRODUCTION

Autonomous mobile robots need to perceive their environment to be able to operate. Typically, a depth sensor is used to create a map of the environment for planning and locomotion tasks. Depth sensing in general and depth from stereo in particular is a well studied subject in the field of computer vision. Measurements are combined with knowledge about the rover pose to temporally fuse depth maps into an environment fixed model.

Since depth from stereo is the central element of the traversability pipeline, it is described here. In particular the derived sensor model will be used for map formulation and analysis in following chapters.

2.2 VARIANCES

Many of the mapping approaches presented in this work rely on statistic principles and estimation. Mathematically speaking, the variance is the expected squared divergence of the samples from the mean. Being a figure of dispersion, its value describes the spread of samples relative to their mean.

For mapping or measurement applications in general, this work distinguishes between two types of variance:

1. Measurement / Sensor Variance
2. Sample Variance

The *measurement variance* σ_m^2 describes the uncertainty of the sensor itself. Based on the sensor model, a certain amount of noise is expected for a given measurement. The described methods will simplify this noise to a Gaussian distribution with the mean being the measured value and variance estimated based on the sensor model. The measurement

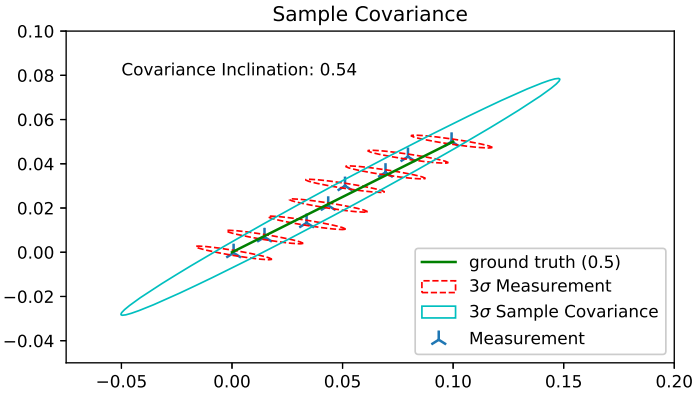


Figure 2.1: Measurement and Sample Covariance Visualization.

variance is attached to each individual measurement and does not have any implication on the ground truth itself.

The term *sample variance* σ_s^2 is used when multiple measurements get statistically evaluated. Both, the sensor induced variance and ground truth value variance σ_{gt}^2 are affecting the *sample variance* (2.1). When bucketing measurements over a discrete parameter region, ground truth variance and therefore sample variance can provide useful insight into the ground truth properties.

For mapping in particular, the difference of measurement and sample variance are exemplarily depicted in Figure 2.1. In the two-dimensional case, the measurement and sample variances are 2×2 covariance matrices. They are depicted as their 3σ equal-probability lines (ellipses).

$$\sigma_s^2 = (\sigma_m + \sigma_{gt})^2 \tag{2.1}$$

2.3 DEPTH SENSORS

Depth or range sensing in general typically utilizes one of two techniques: triangulation or time of flight (tof). Besides those two main methods some application specific techniques are used as for example reflected intensity, inductance and contact based approaches.

2.3.1 *Time Of Flight*

As the name indicates, tof based methods measure the runtime of a transmitted and reflected signal. Based on the signal speed and runtime, the reflection distance can be retrieved. The Measurement accuracy depends on the signal medium together with the timing resolution.

Direct tof methods using light require a very high precision clock to achieve sufficient resolutions for robotic applications due to the high speed of light in air. Since timing precision is typically not dependent on the measured duration, tof techniques can in theory produce measurements with distance independent precision and variance. Typically, direct tof sensors have a reduced precision for close distances compared with triangulation based approaches, but show a significant advantage at medium to high distances. The exact trades are dependent on many system parameters and therefore not generalizable in the context of this work.

A single range measurement is usually not useful for robot mapping and navigation. Sensors used for these applications typically generate a depth image, dense point cloud or two-dimensional scan. This is done by either sweeping a single beam over the scene or by simultaneous illumination and subsequent sampling. Typically, scanning based Light Detection And Ranging (LIDAR) scanners have an operating range of 0.3 m - 100 m with an accuracy of 5 - 10 mm [16]. By adjusting the resulting point density and coverage, full envelope update rates of multiple minutes per scan to several hertz are possible.

Flash LIDAR sensors use a detector array similar to a camera and simultaneous illumination of the scene. A Higher scan repetition rate of up to 60 Hz comes at the cost of significantly increased complexity and price compared with conventional single beam scanning LIDARs. Due to the decreased angle resolution and significant costs, this technique is not commonly used.

2.3.2 *Triangulation*

Triangulation based sensors are less complex than tof sensors regarding the actual measurement method. Distance to a target is determined by intersecting two rays connecting two or more sensors with the same target point. Depth is measured by determining the angle of arrival from the connecting vectors with the known sensor orientations. The sensors are separated by a known [1] baseline b. Two incidence angles and the distance between the anchor points form a well-defined triangle from which the depth can be calculated. More base angle measurements can be used to reduce the depth noise.

Since the angle to depth relation is highly non-linear, measurement variance is depth dependent. The experienced depth variance is dependent on angle measurement variance and the baseline between the two anchors.

1 or in case of structure from motion on the fly calculated

For mobile robot applications typically stereo cameras, structured light cameras and structure from motion cameras are used within the triangulation domain.

While simple stereo cameras track natural texture features in both cameras, structured light cameras add those features to any surface by projecting a known non repeating pattern into the scene. Usually, this reduces measurement noise and improves sparsity in low texture scenes. This comes at the cost of additional power requirements for the projector. Additionally, the outdoor performance of such systems might be lower due to the high intensity broad spectrum light emitted by our central star.

Structure from motion deploys the same feature matching techniques as regular stereo cameras, but does not have a fixed baseline or second camera. This technique relies on the motion of a single camera to reconstruct both: camera trajectory and depth. By solely using image data, a correct depth profile can be reconstructed, but the scale of the scene remains unknown. Typically, this issue is overcome by adding a simple single beam (laser) rangefinder to the setup, which makes the optimization for camera trajectory and depth map solvable with scale. All of this comes with the benefit of being able to chose arbitrarily large baselines. This can be beneficial for setups where the baseline of the stereo sensor would be too large for the system and laser based depth is too heavy or expensive. Depending on the environment and robot state, the used stereo baseline can be changed on-the-go. Static scenes are favorable with this technique for obvious reasons.

External motion capturing systems such as Vicon [17] or Optitrack [18] Use the same camera based angle measurement but deploy more than two cameras to cover a larger space simultaneously.

2.4 DEPTH FROM STEREO

depth from stereo in general describes the process of using two cameras observing the same scene to estimate a depth map. We distinguish between two different types of stereo based depth estimation:

- sparse stereo
- dense stereo

Both techniques use two camera images with slightly different perspectives to reconstruct a depth map. The approaches differ greatly in computational cost and density of the resulting depth map. For both methods the intrinsic and extrinsic parameters of the cameras need to be known.+

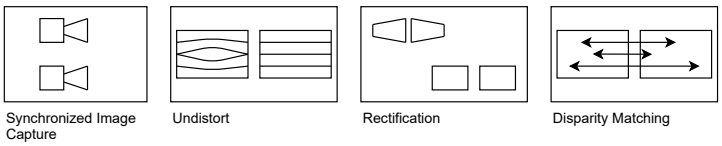


Figure 2.2: Four Steps of Stereo Processing.

Before starting the actual stereo processing, image distortion from the lens and sensor need to be removed. Depending on the extrinsics (how the cameras are oriented to each other) the images are then rectified for having common horizontal equipolar lines. Only the last step, which is responsible for calculating the disparity values differs in sparse and dense stereo.

Sparse Stereo utilizes mathematical feature descriptors to match points between the two images. Correspondences are calculated, and the features are associated with their disparity value. Density of the resulting depth image is greatly dependent on the used feature technique and matching performance. Feature based sparse stereo is usually only used when little computational power is available or when features over different camera perspectives should be tracked.

Dense stereo algorithms on the other hand generate a depth estimate for each pixel. A large number of algorithms have been developed to find stereo correspondences in two images. [19] provides a good overview of the different used techniques. The presented approaches all use calibrated images and determine a pixel disparity value from the left to the right image. With known camera parameters, this value can be used to calculate the depth in the camera coordinate frame.

The four basic steps of stereo processing are visualized in Figure 2.2.

2.4.1 *Intrinsic Calibration and Dewarping*

For a successful stereo processing, lens induced warping and stretching of the image needs to be removed. This can be done by calibrating the camera with a known target and then inverting the determined camera model. Different models with varying level of fidelity can be used for this job. Kalibr [20] is the current research standard for single and multi camera calibration. Supported Camera Models are pinhole camera model [21], omnidirectional camera model [22], double sphere camera model [23] and extended unified camera model [24].

The most simple pinhole projection model only estimates the focal lengths and image center coordinates. Additional distortion models can be applied to undistort lens effects which supersede the camera

model itself. Those distortion parameters model the manufacturing imperfections rather than conceptual coefficients.

The for this work used camera model is called CAHVOR [25, 26]. Each of the letters represents a distinct set of parameters:

- C: Camera Center Vectors
- A: Camera Axis Unit Vector
- H: First Image Plane Vector
- V: Second Image Plane Vector
- O: Optical Axis Unit Vector (used for lens-distortions)
- R: Radial Lens Distortion Coefficients

CAHVOR calibration is performed with a JPL internal tool. The JPL stereo library requires a CAHVOR camera model for operation.

2.4.2 *Rectification*

The rectification step takes care of potentially unmatching equipolar planes. Stereo matching requires, that pixels associated to the same point in 3D are lying on the same camera scan line. This way, instead of searching through the entire image for matching correspondences only a single row needs to be evaluated.

Rectification is done by warping the image from the second camera based on previously determined extrinsic parameters. Naturally, only the overlapping area between the two images can be evaluated for disparity.

2.4.3 *Dense Stereo Matching*

For each pixel in the left image, the corresponding pixel in the right image needs to be identified. Due to the previously applied rectification, this pixel must be on the same row. Recovering this per pixel disparity can be done with many techniques. All dense stereo matching approaches assume certain simplifications of the real world. For example, intensity based matching usually assumes Lambertian [2] surfaces.

A small image patch centered around the pixel in question is used for template matching. The maximum similarity for left and right images is then chosen for the disparity pair. Various methods are used in literature

2 Surface look the same irrespective of the viewing angle.

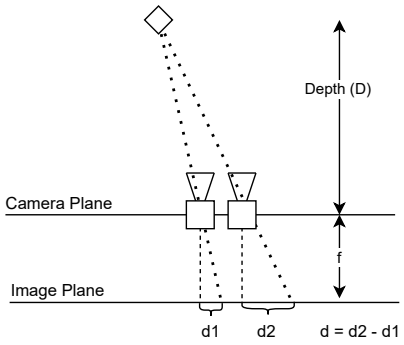


Figure 2.3: Trigonometry for Depth Calculation

to express similarity of two image patches of same size. The two most common intensity metrics are:

Sum of squared differences (SSD):

$$SSD(img_L, img_R) = \Sigma_x \Sigma_y \left(I_{img_L}(x,y) - I_{img_R}(x,y) \right)^2 \tag{2.2}$$

Sum of absolute differences (SAD):

$$SAD(img_L, img_R) = \Sigma_x \Sigma_y |I_{img_L}(x,y) - I_{img_R}(x,y)| \tag{2.3}$$

2.4.4 *Sensor Model*

In the last step, the acquired disparity in pixels must be converted to depth values. Due to the known setup, this is fairly easy and simply a calculation based on Baseline (B), Focal Lenght (f) and disparity (d). Figure 2.3 shows the setup and which elements affect the disparity. Depth is calculated by:

$$z = \frac{f \cdot B}{d} \tag{2.4}$$

Since depth is inversely proportional to disparity, the perceived depth variance caused by image noise must be modeled explicitly. Figure 2.4 illustrates the relation.

Precise identification and modelling of stereo noise can be an entire thesis topic on its own. A wide range of literature covers this topic [27, 28] and introduces noise models from simple distance dependent normal distributions to complex models based on the intrinsic parameters of the camera. For simple mapping tasks such as the one presented in this thesis, one of the simple models has shown to be sufficient. The depth error from stereo is estimated with a three-dimensional covariance. This covariance has the shape of an ellipsoid with one major axis and two

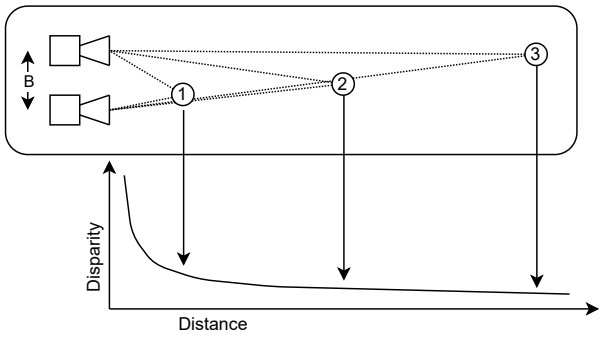


Figure 2.4: Inverse Proportional dependency between depth and Disparity.

nearly similar dimensions. The major axis is aligned with the camera vector and resembles the direction of maximal variance. Perpendicular directions have a much lower variance. Variance calculations for the individual axis are presented by [29].

In the following, d represents disparity, (u, v) is the pixel position in the image relative to the image center, B is the stereo baseline and f is the cameras' focal length. Two error parameters influence the shape of the covariance matrix: Pointing error or pixel size (p) and matching error (m). The matching error describes to which precision the stereo matching algorithm is capable of determining the disparity, while p is dependent on the camera calibration precision. Error functions for X and Y are quite simple due to their sole dependence on p:

$$\delta_x = \frac{p \cdot z}{f} \tag{2.5}$$

$$\delta_y = \frac{p \cdot z}{f} \tag{2.6}$$

Deriving the depth by the disparity yields:

$$\frac{\delta z}{\delta d} = -\frac{f \cdot B}{d^2} \tag{2.7}$$

$$\delta z = \frac{-z^2}{f \cdot B}\delta d \tag{2.8}$$

Since m is the uncertainty in disparity, it will be substituted for δd. The resulting normal distribution is then modeled as the measurement itself being the mean with a covariance $\Sigma = Diag(\delta_x^2, \delta_y^2, \delta_z^2)$.

2.5 POSE VARIANCE PROPAGATION

Pose Variance Propagation is used to handle the uncertainty in robot localization during mapping. After movement of the rover, old mea-

surements stored in the map are not any more known to the same uncertainty of the original measurement. Pose drift and noise in the localization moves the robot relative to previously observed regions, thus degrading their known precision. Modeling this effect requires a good understanding of the localization noise characteristics. Since localization and navigation is beyond the scope of this work, we assume localization noise to be known. [30] has formulated a pose noise consideration into the system update of a Kalman filter based map. The presented variance propagation builds on this work.

The certainty of each measurement is modeled as a three-dimensional covariance matrix. It does not matter, if single measurements are considered or variances are formulated in terms of grid cells. At observation time, the uncertainty of the map cell equals the measurement uncertainty. In the lateral direction, this remains true as long as the cell is actively observed. [3] Unobserved elements degrade in position certainty by the accumulated and projected pose uncertainty of the rover.

J_r is the Jacobian with regard to position uncertainty, J_Φ the Jacobian regarding orientation uncertainty and J_m the Jacobian regarding the previous state. Σ is the respective covariance matrix. The propagation of uncertainty calculates as following:

$$\Sigma_{m,t} = J_m \Sigma_{m,t-1} J_m^T + J_r \Sigma_r J_r^T + J_\Phi \Sigma_\Phi J_\Phi^T \tag{2.9}$$

To illustrate how the uncertainty distribution of a cell is affected by the three sources of pose variance a simple experiment is conducted: The following base covariance is used:

$$\Sigma_{base} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{pmatrix} \tag{2.10}$$

Figure 2.5 visualizes the lateral components (x, y) through a probability surface and equiprobability lines.

2.5.1 *Sources of Variance*

The effect of each of the sources of pose variance is illustrated.

2.5.1.1 *Different Sub Map*

Typically, consecutive sub maps are rotationally aligned, which yields the identity matrix I for the Jacobian J_m. For the sake of presentation,

3 Some variance estimating algorithms might update the height variance independently.

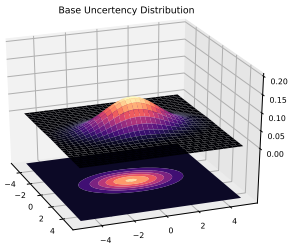


Figure 2.5: Base Uncertainty of the Simulated Cell.

we assume a 30° yaw rotation between two sub maps[4]. Since no new source of error is introduced, the already known covariance is simply rotated with the vehicle frame as depicted in Figure 2.6.

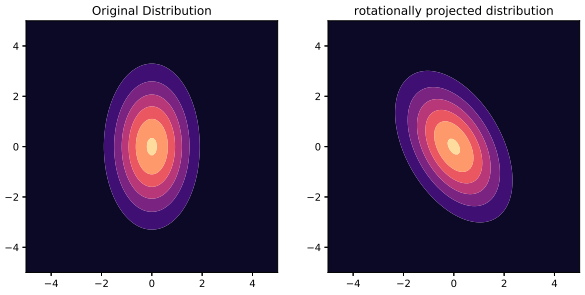


Figure 2.6: Rotated Covariance Due to Rotation in Successive Map Frames.

2.5.1.2 *Uncertainty in Translation*

Translational uncertainty accumulated by movement without observing the cell in question is an additional source of variance. The depicted additional variance is:

$$\Sigma_r = \begin{pmatrix} 0.1 & 0 & 0 \\ 0 & 0.15 & 0 \\ 0 & 0 & 0.2 \end{pmatrix} \tag{2.11}$$

For the sake of presentation, a rotation of 20° yaw, 15° pitch and 20° roll is assumed to have happened in between measurement time of the cell and current time. Its original, additional and resulting uncertainty are depicted in Figure 2.7.

4 This happens, when a robot centric map is rotation locked to the vehicle instead of a global reference.

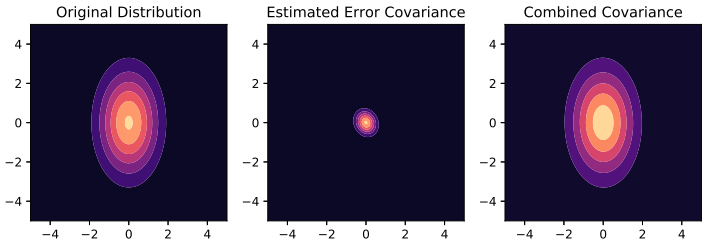


Figure 2.7: Covariance With Additional Position Uncertainty.

2.5.1.3 *Uncertainty in Rotation*

Finally, this source of variance deals with the uncertainty introduced by an erroneous yaw angle estimation. The other two rotations are already covered in the measurement step, since they are of elementary importance for the height uncertainty. Yaw uncertainty effects lateral covariance the most, thus it is handled here. Since lateral uncertainty projection of a yaw error is strongly dependent on the distance and direction from the robot to the cell in question, the covariance for a cell at the position $\begin{pmatrix} 1 \\ 5 \\ 0.5 \end{pmatrix}$ relative to the robot is depicted. The rotation covariance matrix is

$$\Sigma_\Phi = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0.1 \end{pmatrix}$$

The resulting change in uncertainty is depicted in Figure 2.8

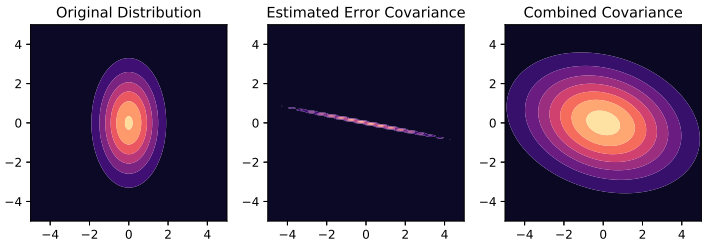


Figure 2.8: Covariance With Additional Yaw Uncertainty.

2.5.2 *Efficient Propagation of Variances*

In theory, the propagation of pose variance has to be performed every time the map receives a robot pose update. All cells get degraded by the translational and rotational uncertainty update. New measurements

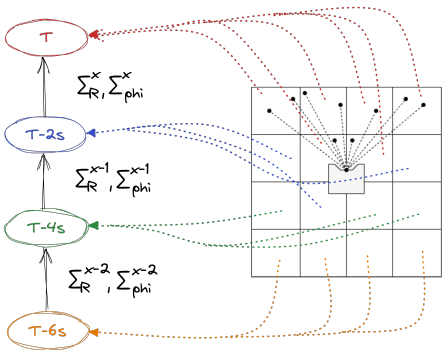


Figure 2.9: Uncertainty Graph Structure

in the consecutive measurement update pull the variance of freshly observed cells back to the measurement variance.

Since the variance values of each cell are only used upon map query or new measurement insertion, variance propagation can be delayed to those events. Not every cell has to be queried or updated at the same time. This cell specific pose noise propagation can easily be done with a structure as shown in Figure 2.9.

A helper structure is introduced, the so-called *history uncertainty graph*. Within this directed graph a node is added for each pose update of the rover (e.g. after a certain distance, until accumulated uncertainty overshoot a threshold...). Each cell has a memory pointer to one node of the *history uncertainty graph*. Every time, a cell is measured (and thus covariance is updated) the cell gets a pointer to the latest *history uncertainty graph* node. On each vertex of the graph, the respective accumulated variance in position and yaw are stored. Those values are already projected into the map frame. Essentially, $J\Sigma J$ is stored on the edges.

Upon cell query the memory pointer is followed into the graph and covariance can be accumulated by addition along the vertices. Finally, the propagated covariance matrix is used for updating the covariance of the cell and the graph pointer is pulled to the latest node. Nodes with no incoming pointers can either be collapsed with its successor or deleted, if the node in question is the tail of the graph.

With this simple structure, no map operations are necessary when the rover moves. Accumulation of pose uncertainty is bundled and calculated in a highly optimized manner. Consideration of lateral pose uncertainty with this technique comes at essentially no additional cost.

3

STATE OF THE ART MAPS

Within this chapter the state of the art of mobile robot mapping is outlined. The main emphasis lies on the actual terrain representation of the different systems. This chapter focuses on approaches, which might be feasible implementing on space hardware.

Mobile robotic platforms typically use one or more depth sensors to perceive their environment in three dimensions. Even though some approaches [31, 32] directly use the sensor data for path planning, usually temporal fusion is applied to reduce noise and add history to the representation. Local maps are a temporal fusion of previous measurements. New measurements change the representation in an incremental way.

3.1 INTRODUCTION

Different methods of representing terrain for mobile robotics are well documented in the body of literature. They differ greatly in complexity, fidelity and capabilities. Application tailored approaches usually consider the type of encountered terrain, sensor parameters and computationally constraints.

Usually, terrain mapping in mobile systems is used for some kind of traversability analysis and planning. The terrain representation can either explicitly store identified hazards or is evaluated to recover the obstacle representation. Terrain traversability analysis and surface mapping is covered by a great literature basis. Despite this attention the same process is known under the following keywords according to [33]:

- drivability
- trafficability
- navigability
- coverability
- terrainability
- maneuverability

- mobility
- traversability

Within this thesis the term *traversability* is used.

3.2 TERRAIN REPRESENTATIONS

We first have to distinguish between 3D, 2.5D and 2D map representations for mobile robotics. While full 3D representations allow for the most flexibility of the deployed system, its drastically increased processing and memory requirements can make it unfeasible for some systems. Particularly in a Lunar environment, no overhanging structures are expected or need to be traversed. As for the Lunar application, 2D or 2.5D maps are sufficient for many mobile robotic systems.

In many approaches, mapping is carried out in a robot centric way. Hereby, the terrain immediately surrounding the robot is stored and moved together with the robot.

While a simple shifting window around the robot is a simple implementation of this for grid based approaches, in particular mesh or surfle based maps need continuous re-meshing for moving the map [34]. Other techniques such as the famous Octomap [35] , Voxblox [36] or even simple Quad-Trees [37–39] are constrained in efficient map movement by their used data structure. Continuous copying of data ensures robot centricity if needed.

In general, applying a map in a global way usually does not imply additional considerations.

3.2.1 *2D - Occupancy Grid*

The most basic form of mobile mapping is the simple 2D occupation grid as presented by [40, 41]. This approach simply discretizes the surrounding world in squares of equal size. Each cell can either be occupied, free or unexplored. An exemplary visualization of a simple 2D occupancy map is shown in Figure 3.1. The state of the cell is determined with the assumption, that the presence of measurements within the cell indicates occupation. Each cell is modeled as an independent random variable. Used as a foundation, many other works enhanced occupancy grid mapping during the early days of mobile robotics by introducing new update algorithms [42–44]. Most prominently two update algorithms emerged. The by [40] demonstrated Heuristic Probability Function Mapping uses a probabilistic sensor model to update the probability of a cell being occupied on a turn by turn basis. Histogrammic in Motion Mapping, introduced by [45] updates only a single cell for each sensor

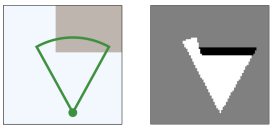


Figure 3.1: Simple 2D occupancy map with ground truth. White = traversable, Black = obstacle, gray = unknown [47]

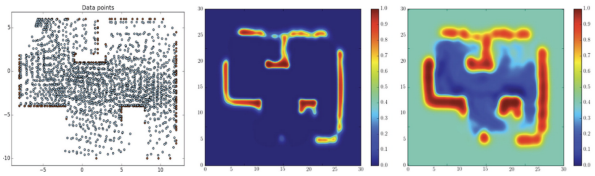


Figure 3.2: Visualization of Hilbert maps [48].

reading. The probabilistic distribution is hereby obtained by rapidly sampling the sensor while the robot is in motion. Both approaches are benchmarked by [46] with Histogramic in Motion Mapping turning out as the superior technique. These maps are created with a simple two-dimensional range sensor operating in the plane of motion. Higher dimensional sensor data can be post-processed for insertion into these mapping algorithms.

A different take on 2D Occupancy Maps is the Hilbert Map presented by [48]. Instead of modelling the world as a discrete space of random variables a continuous space of occupancy is created. The arbitrary resolution offered by Hilbert Maps is archived by projecting a high dimensional feature vector into the Hilbert space. A visualization of a Hilbert map is shown in Figure 3.2.

Even though simple 2D Occupancy Grid Maps are the earliest types of mobile robotic maps, their usage in state-of-the-art systems is unrestrained. Most ground born robot path planning algorithms rely on some sort of binary or single dimensional traversability map [49]. Those traversability / occupancy maps can be computed from different inputs as global point clouds [50] or directly from the depth sensor data as part of the processing pipeline [51].

[52] presents a fuzzy drivability estimate based on slope and roughness. Such linguistic terms can either be stored in form of a grid map or directly used for navigation.

To a certain extent the idea of pre-defined cells being either occupied or unoccupied can be extended to the third dimension. [53] Has demonstrated this by using stereo photographs and a 128x128x128 voxel grid.

Instead of saving the occupation value in a regular grid, it can be saved in an adaptable data structure. Quad-Trees increase there resolution at obstacle borders while using less memory for fully open or occupied

space [37–39]. This data structure makes multi resolution representations straight forward, but complicates the application of robot centric mapping.

3.2.2 *2.5D - Height Map / Piecewise Constant*

Similar to the *traversability* terminology different names for the grid based 2.5D terrain representation are known:

- height map
- elevation map
- DEM
- terrain map
- Cartesian elevation map

Originating from occupancy grids, grid structures can easily be extended from occupancy to a terrain model by attaching a height value to each discretization cell [54, 55]. These grid based terrain representations are computationally and memory efficient while allowing for optimized path planning algorithms based on graph search. They can easily be transformed to binary traversability (occupancy) maps by applying simple filters to the map. One early height map post-processing approach is presented by [56]. Hoffmann and Krotkov use multiple spatial Fourier analysis to extract roughness parameters from the height map. Another common and even simpler approach is the roughness estimation based on a sliding window of predefined size. [57] uses a predefined kernel shape and determines the maximum height difference within this shape. By executing this operation for each cell a roughness map is created. Cells which surpass a certain roughness value are considered non-traversable.

[58] uses a grid map which stores mean height and confidence values. The confidence increases linearly with the number of added points. Traversability is estimated by calculating the minimum of local slope and roughness. A simple motion planner is added by evaluating the local traversability map in cones emerging from the map center. Each cone counts the number of non-traversable cells. The cumulative traversability (over bearing) is then plotted as a histogram over the full circle and the direction with the least amount of obstacles is selected.

Traversability estimation is not restrained to only use map data. [59] for example combines LIDAR based terrain analysis with image analysis to estimate the traversability of terrain patches. These values are then attached to a similar map structure.

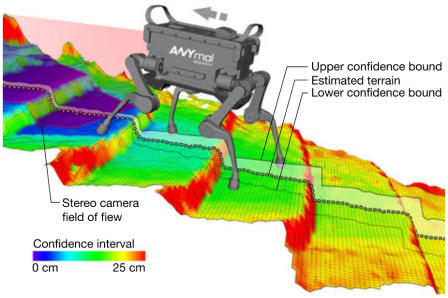


Figure 3.3: 2.5D Robot Centric Elevation Mapping Framework [60].

When it comes to the pure update of integrating a set of new measurements into the existing map different techniques can be applied. Most commonly, a simple Kalman filter update is used to estimate the mean height based on predicted measurement variance. [30] presents a Bayes / Kalman based height map for a mobile robotic system. The elevation map is robot-centric and estimates the Kalman Variance in Height direction based on the sensor model and localization noise. How the robot motion in particular can affect variance estimation is described by [60]. At the time of writing this technique is actively applied in many state of the art robotic systems [61] due to its simplicity and reliability. 3.3 shows the generated local terrain abstraction.

A similar height map update is presented by [57]. It extends the simple Kalman Update by correctly estimating the sample variance for a better understanding of sub cell features. With one extra value per cell, a correct upper and lower bound of the terrain can be estimated. This technique is further referred to in this thesis as OMG and is extensively analyzed.

If the encountered terrain contains overhangs or even bridges, the simple height grid can be extended to incorporate multiple starting height layers as done by [62]. This way, the robot is able to perceive passages, which are traversable, but covered. Additionally, loop closure is introduced for these kinds of maps.

Being the most common technique, Piecewise Constant grid maps are applied in many more robotic applications and are continuously adapted to specific requirements.

3.2.3 *2.5D - Mesh / Piecewise Linear*

The assumption of a continuous terrain can be used as a prior to formulate mapping updates for mesh like terrain representation. Each node represents a certain height in the world. Space in between nodes is linearly approximated based on the three closest nodes. More than other approaches, this incorporates a smoothness prior during terrain

estimation. Multiple resolutions of the same mesh can be represented by re-meshing the same mesh for certain regions. Since measurements typically do not coincide with only one node, incremental map updates are difficult to realize efficiently.

[63] presents an application of triangular piece wise linear maps for robotic navigation. By using conventional height updates, the mesh remains regular in the lateral dimensions. Barycentric coordinates are used to weight the mesh updates as input for the optimization problem of incorporating new measurements into the existing map. Multiple resolutions are implemented by local tessalation of the mesh.

[64] extends this mapping technique by introducing a linear time measurement update and height variance awareness. By attaching a variance to each surfle, not only height uncertainty is modeled, but implicitly also slope variance can be retrieved. Rover localization noise is mitigated by tying some nodes of the mesh to identifiable landmarks in the real world. Therefore, no variance degradation of old measurements is necessary for a correct representation.

3.2.4 *2.5D - Stochastic Map*

These mapping approaches associate not only one height value to each discretized cell but also consider statistics within the cell. This can range from a simple cumulative weight over variances in the Z-dimension to inclination in two axes, roughness and additional variances as presented by [65, 66]. Those stochastic map representations have their roots with [67] from 1994. Within these applications, estimated slope, roughness and confidence can be threshold and used as a binary occupancy grid. Stochastic parameters are estimated by plane fitting and adaptive weighting of measurements. Many approaches are used for cell wise parameter estimation and fusion. The in this thesis introduced CCM algorithm can also be categorized here.

Following a similar idea to [65] sub cell information can be stored in the parameters. Traversability estimation ultimately can be tied to both, inter- and intra-cell analysis. Stochastic cell mapping can also apply to 3D as shown by [68, 69]. Three-dimensional covariance described ellipsoid surfels are fitted to *occupied* 3D cells for increased precision. [69] augments this 3D mapping with a technique for improved map alignment based on those probabilistic surface representations. Higher precision and improved runtime over Iterative Closest Points (ICP) are acheived by the presented surfle matching technique. Figure 3.4 shows the 3D application of covariance based surfle mapping.

A more system focussed approach to height map based traversability estimation is presented by [66]. For each cell, the roll and pitch an-

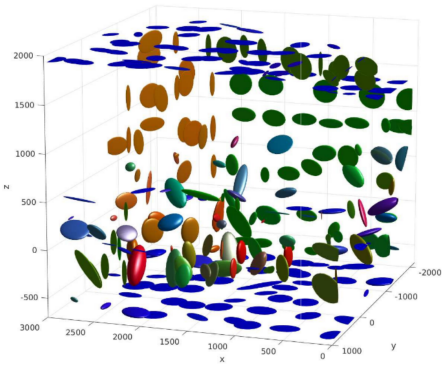


Figure 3.4: Ellipsoid Surfle Map with from Covariance Fitting [69].

gle is estimated additionally to the roughness by simple least-squares plane fitting. Those metrics are estimated directly on the stereo point cloud. Contrary to other map implementations, this technique does not incorporate a height representation and uses the fit residual as the roughness parameter. Slope and roughness are directly evaluated to attach a drivability score to the cell for further motion planning.

This completes this surveys coverage on 2D maps. For a better overview of different traversability estimation techniques please refer to [33] for a comprehensive survey on traversability estimation techniques.

3.2.5 *Surfel*

Especially in the field of 3D Visualization surfel maps are a commonly used surface representation. Similar to 3D Voxel map or the previously described ESM, a discretized space is represented by attaching a state to each cell. Surfel-based techniques chose from a set of surface elements (Surfels) [70] to increase the fidelity of the map. Not only if a Voxel is occupied or not, but which parts of it is therefore part of the representation. Surfel maps are used in Robotic and SLAM applications [71, 72]. Despite producing visually pleasing representations their usability for obstacle avoidance remains rather limited. Without enforcing closed surface meshes, distance queries and thus obstacle avoidance is difficult to accomplish.

3.2.6 *Signed Distance Fields*

Signed distance fields are equally common for 3D as for 2D applications. Volumetric SDF based maps [36, 73, 74] provide the distance to the closest surface for each point in space. In two dimensions by using SDFs for hazard maps, a distance to the closest obstacle can be queried directly. Especially for path planning algorithms, this representation is useful. Incremental updates by fusing new depth measurements into

the map are unfortunately not straight forward. Additionally, SDF maps are usually global representation without the ability of moving the map together with the rover to form a local representation.

Local mapping can be achieved by splitting the local and global map as introduced by [75, 76]. The moving agent creates small local SDF maps based on sensor data, while a pose-graph solver aligns sub maps to form a global representation.

3.2.7 *Machine Learning*

More modern techniques try to combine the terrain analysis and planning into one consolidated neural network [32, 77]. Map information is essentially encoded into the model through training. Even though promising results are presented, the increased resource utilization is not justified by the performance yet.

A different approach to modern mapping techniques is the augmentation of conventional geometric traversability maps with data from semantic segmentation as shown by [78]. With pure geometric analysis, non-rigid features such as soft bushes are classified as obstacles [61]. By combining image based semantic understanding with the mapping pipeline, potentially hazardous terrain can be identified more reliable. For planetary robotics, [79] shows the potential of this technique for soil adaptive planning on Mars.

Learning based traversability analysis as presented by [80] can be applied to conventional 2.5D Elevation maps for more precise classification. In particular, when the robots' mobility is capable of traversing certain structures (e.g. stairs) a learning based classification brings the benefit of identifying known traversable features in a non-geometric way.

4

MULTI SIZE MULTI RESOLUTION MAPPING

This chapter discusses the developed mapping framework and reasons about the design decisions. Since the presented software architecture will be part of CADREs' flight software, system parameters of the final rover are a major design driver and are also presented in this chapter.

Generally, a regular grid based map structure is used.

4.1 GENERAL MAP SETUP AND REQUIREMENTS

The major design driver for the framework is the with squared distance increasing stereo measurement error. This combined with the inverse square law governed decreasing point density make a single resolution map unfeasible. Either, the map would suffer from severe sparsity and measurements associated to a wrong cell or the highest resolution would not be able to satisfy the needs of the motion planning subsystem. All these factors motivate a multi size multi resolution map, where high resolution layers surround the robot with a smaller footprint and low resolution layers cover a wider area. Size and resolution of individual map levels can be motivated with the expected stereo variance at certain distances. The high resolution layer in proximity to the rover needs to resolve all drive hazards including small obstacles and slope violations. Even though the used stereo camera is incapable of resolving small obstacles at a distance, the low resolution layers still detect big obstacles and slopes. They can be used for rough path planning and team-level coordination.

Due to limitations in the available memory, only a small terrain patch can be stored at a time. Consequently, the map has to move with the rover to stay useful. To preserve terrain features, cell borders need to stay fixed with respect to the surface. Robot centricity of the map is achieved by moving the map once the rover traversed the distance of one cell. All map layers must move individually to keep high resolution layers centered. Instead of copying the entire map content each time the map moves, a rolling buffer uses dynamic indexing for map movement. Only cells which leave the covered area are cleared and assigned to the freshly covered space in the direction of movement. Rolling the map reduces the computational overhead significantly.

Finally, the map needs to support pooling operations. This way, new measurements only need to be added to a single layer when performing a measurement update. Upon map query, new measurements get propagated to the other (lower resolution) layers to keep the map integrity intact. Because of the individual map movement, cell correspondences between layers are not permanent. The map architecture needs to provide a way to dynamically find *daughter* cells efficiently to support the pooling operation.

Selection of the numeric values for sizes, resolutions and filters is based on the parameters and requirements. The exact numeric selection of these values is highly dependent on the system setup. An exemplary design is later presented in Chapter 7.

For testing purposes, the framework must support Robot Operating System (ROS) besides the flight software F' interfaces without changing any core part of the software. Initial software setups have to support different mapping updates. After decision of the optimal map update, other updates must be removable residue free.

4.2 MAP ARCHITECTURE

4.2.1 *Laplacian Pyramid*

To satisfy the requirement of supporting multiple resolutions with individual dimensions a map structure needs to be deployed, which supports seamless switching between layers. The core concept of multi resolution maps presented in this work is loosely based on the work of [63].

Essentially, multiple individual maps with different size and resolution exist at the same time with overlapping footprint. The different resolution maps are referred to as layers or levels.

Map layers in this setup have common cell borders. This way, each low resolution cell is associated with $4^i, i \in \mathbb{N}$ child cells. [1] By employing cell subdivision it is guaranteed that a low resolution cell footprint corresponds entirely to a set of high resolution cells. This is a necessary requirement for successful pooling. Since each level contains a surface representation of different scale, it is important to maintain consistency between the layers by either updating all resolutions independently at update time or utilize pooling at query time.

The layer stackup is visualized in Figure 4.1. Total Physical dimensions of the low resolution layers are greater than the high resolution ones as shown in Figure 4.2.

1 As long as the same area is covered by the high resolution level.

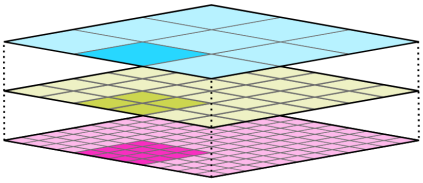


Figure 4.1: Map Cell Level Correspondence. Source: [81]

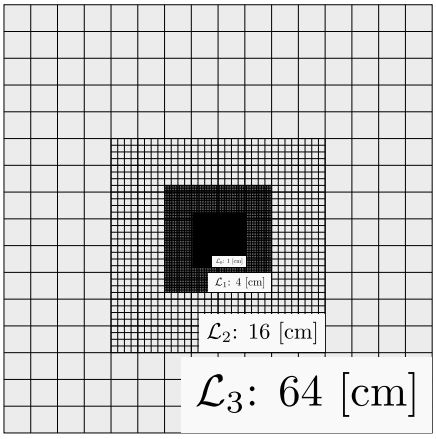


Figure 4.2: Multi Size Robot Centric Setup

4.2.2 *Map Pooling Vs. All Layer Insertion*

Consistency over the redundant (overlapping) portions of the map can be achieved by either updating all corresponding cells on all levels during the measurement update or by performing a dedicated pooling operation. While naively performing the same update for all levels is an easy and straight forward way, it also greatly effects the runtime of the measurement update.

Pooling is performed on query time. Therefore, consistency is only enforced when it is actually necessary without computational overhead. During the measurement update, new measurements are only inserted into the highest resolution available for the particular location. Pooling then propagates those updates to the low resolution layers. Only one operation per cell is needed instead of one operation for each new measurement.

While individual updating performs the same operation for each measured point for each layer, a pooling based setup only runs the measurement update once and then one update per cell. The exact algorithms fusing multiple high resolution cells with the prior from the low resolution cell into a new posterior low resolution cell is specific to the used mapping algorithm. For example in simple piece wise constant maps, high resolution cells can be treated as individual measurements for the low resolution update. The information flow is visualized in Figure 4.3.

After pooling, data in the cells is identical to the result obtained from all layer insertion.

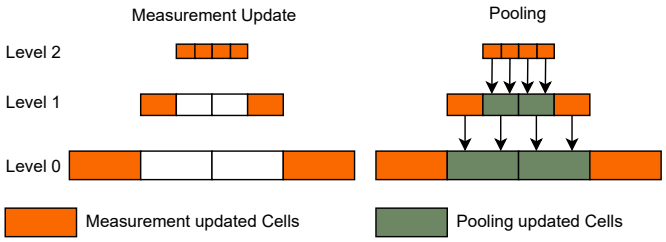


Figure 4.3: Information Flow for Measurement and Pooling Update.

4.2.3 *Rolling*

Keeping the map centered around the robot is achieved by moving the map as soon as the robot traverses out of the central four cells. This operation can either be performed for each layer individually (independent map movement) or once the robot traverses the base resolution (dependent map movement). Independent map movement has the advantage of precisely keeping each layer robot centric. Especially when the difference in resolution between base and highest resolution is significant, independent map movement allows for a consistent map coverage in all directions for each map level.

The main advantage of dependent map movement is a simplified pooling operation. Cell correspondences between levels can be fixed and do not have to be re- calculated for each pooling. In multi dimension map setups such as the one presented in this work, dependent movement additionally brings the advantage of never encountering partially covered cells. This can happen, when the high resolution layer has moved an amount of times which is not a multiple of the cell's subdivision. At the border of the high resolution layer, low resolution cells are only partially covered. Statistically valid pooling is not possible for these cells. By simply excluding partially covered cells from pooling, this issue can be mitigated. Border cells are updated by inserting the measurement in both valid layers.

For the presented application the advantages of independent movement outweigh the increased complexity. Only independent movement is therefore considered from here on.

In order to be able to move each map level efficiently, the data storage is implemented as a two-dimensional ring buffer. Data in the overlapping area of pre-and post map movement remains untouched in the memory with no additional copy operations. The *rolling* operation of the map is visualized in Figure 4.4. Cells at the opposite end with respect to the direction of movement are deleted and assigned to the new frontier. In addition to the advantage of efficient memory management, the

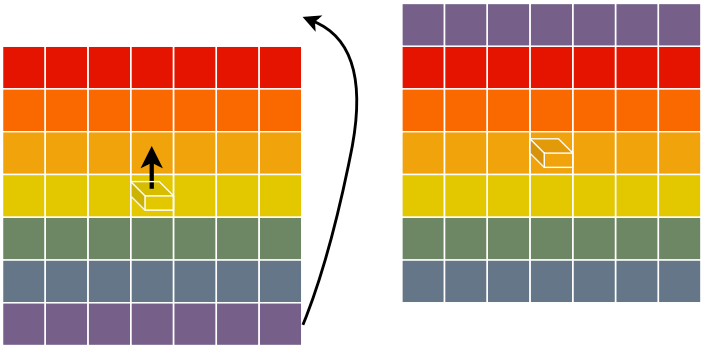


Figure 4.4: Memory Association for Map Rolling.

environment is discretized only at initialization of the map. The cell boundaries do not move after that.

By choosing an even map size, the center of the map is represented by the common corner of four cells. All four cells are selected to be the central element. The map is only moved when the rover leaves those four cells. By having this two by two inflated area no flickering can occur when the robot stops right at the intersection of two cells. This one cell hysteresis prevents unwanted map movement caused by a noisy robot position.

4.2.4 *Software Architecture*

The mapping core is structured in data storage, map update and interface. As a foundation, the data storage class handles all data access operations. Accessors are providing an interface for either position or batch queries. Position queries are typically used for measurement insertion while the batch accessor provides an efficient way to output entire layers.

Data processing itself is handled by the map implementation classes, which inherent from the Map Foundation. New measurements are fused into the map in different ways, while the data structure remains unchanged.

To account for different sources of data during development and testing an interfacing class takes incoming data and calls the right functions in the mapping core. This interface adapts to different middlewares. The complete layout with all major methods is shown in Figure 4.5.

4.2.4.1 *Data Storage*

Being the central element of the mapping framework, the *Map Foundation* class handles all memory access and provides accessing functions. Besides providing an interface for wrappers and update implemen-

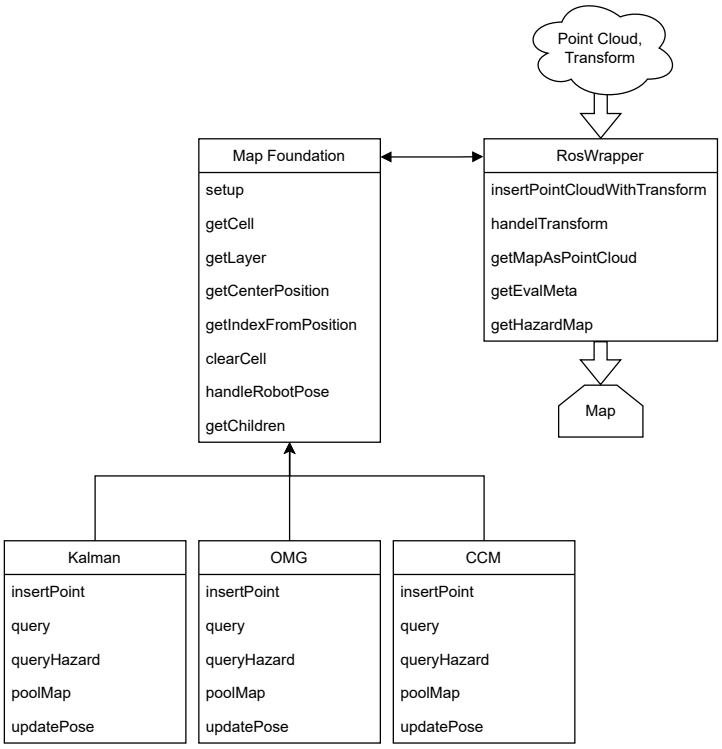


Figure 4.5: UML Diagram of the Mapping Framework Core.

tations, low level operations such as map movement, cell access and queries are taken care of by this element.

The data itself is stored in fixed size C arrays to increase accessing speed and comply with the strict coding guidelines for NASA flight software. Consequently, map sizes need to be selected at compile time.

The *Map Foundation* provides multiple interfaces to this memory. Besides the fundamentally necessary *Get Cell At Position* accessor, functions to support pooling and layer sized batch query are provided.

4.2.4.2 *Map Implementations*

Map Update Classes inherent from the Data Storage class. Here, incoming measurement data is fused into the existing terrain representation.

Based on the implemented algorithm, a location query is performed on the data and corresponding cell values are altered. OMG, Kalman and CCM updates are used for evaluation purposes. These map updates are presented in greater detail in the upcoming Chapters. Using the stored cell data for estimating traversability is also part of the map implementation to preserve adaptability to different techniques.

The flight software map is a mixture of a Kalman Filter and CCM update. Similar to the algorithms implemented for evaluation, it is simply another *Map Update* Class using the same foundation.

4.2.4.3 *Interfaces*

Interfaces or wrappers connect the Mapping Core with other infrastructure. For testing purposes, a ROS interface was deployed, which accepts ROS messages to call the low level API functions of the map.

A second interface connects the mapping core with F′ messages for flight software operations. The mapping algorithms and storage classes remain untouched for this transition.

Specific functions of the wrapper classes greatly depend on the incoming data and user needs of the map. Typical candidates are *Insertion of New Measurements*, *Handle Robot Movement* and *Output the Map*.

5

CONVERGING COVARIANCE MAP (CCM)

This Chapter presents the map update algorithm CCM. After a short description of the motivation behind this update, the algorithm is mathematically derived and presented in detail.

One of the unique features of this map is the implicit surface roughness representation for obstacle detection. Sub-cellsize obstacles can be detected without the need of costly post-processing algorithms.

By estimating the parameters of a 3D measurement distribution per cell, much higher precision maps are acquired at a similar cost to traditional piecewise constant maps. A first order fit approximates the surface in each cell as an inclined surface to distinguish between slope and obstacle traversability hazards.

The technique presented in this chapter will be used for the low-resolution layers of the CADRE flight map and forms one of the main contributions of this thesis.

5.1 MOTIVATION

Conventional piecewise constant maps with a single elevation per cell are a straightforward and computationally inexpensive representation of the terrain. They have a significant drawback when it comes to inclined sections of the terrain. Cells covering a slope will always experience high variance data even when the measurement noise is negligibly low. This is best illustrated in the 2D example:
In Figure 5.1 a simple continuous terrain is measured with a very low amount of noise. The error bars indicate the mean and standard deviation (1σ) in height for each cell.
It does not come to a surprise, that cells covering a high inclination patch only represent the underlying terrain well close to the center of the cell. Measurement spread and therefore perceived variance is subsequently high even though the original data has a high precision.

This effect can be mitigated by either increasing the piecewise constant map resolution significantly, or by estimating a higher order surface representation of the individual cells. As shown later, the former approach reaches its limits due to the decreasing number of measurements per cell and measurement variance constraints. A low measurement count

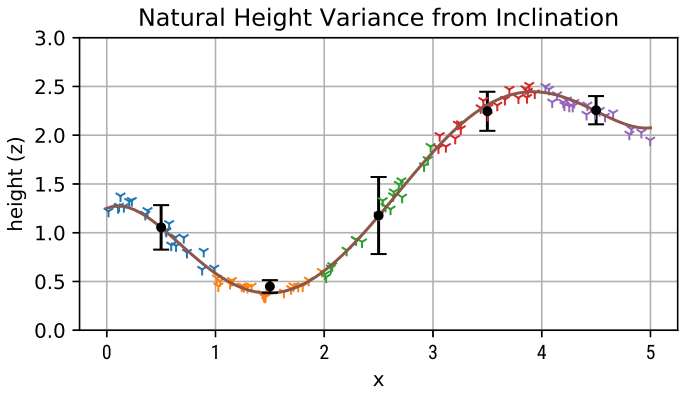


Figure 5.1: One Dimensional Example for Inclination Induced Height Variance in piecewise Constant Maps.

per cell drastically decreases height estimation precision and can even result in map sparsity. Outlier resilience is highly reliant on a high measurement count per cell.

To solve this issue, CCM estimates the full covariance of each cell distribution together with a measurement mean. The variances can be used to reconstruct a first order slope within the cell. If obstacles are present within one cell, the perceived sample variance in the cell normal direction increases.

5.2 OPTIMAL MAP

To understand the choices, which led to the CCM algorithm in its current form, the *optimal* [1] map is presented first. Techniques presented in this Section are not directly used in CCM but give an insight into what would be the optimal 2.5D map fusion. Even though in reality stereo measurement noise does not strictly follow a normal distribution, it has shown to be a valid abstraction. The presented map estimates the measurement uncertainty according to the previously described sensor model from section 2.4.4.

The idea behind this technique is to store every measurement mean with its associated measurement covariance $\Sigma_m \in \mathbb{R}^{3\times3}$, as described in Section 2.2, by its own and forever. When the map is queried, all measurements are evaluated by their likelihood of being at the queried x / y coordinates parametrized with height. Essentially, this operation slices all three-dimensional measurement distributions at one two-dimensional point. Gaussian inference is used to combine all resulting one-dimensional (height) likelihood distributions. The most likely

1 as in all available information is considered and nothing gets simplified or merged

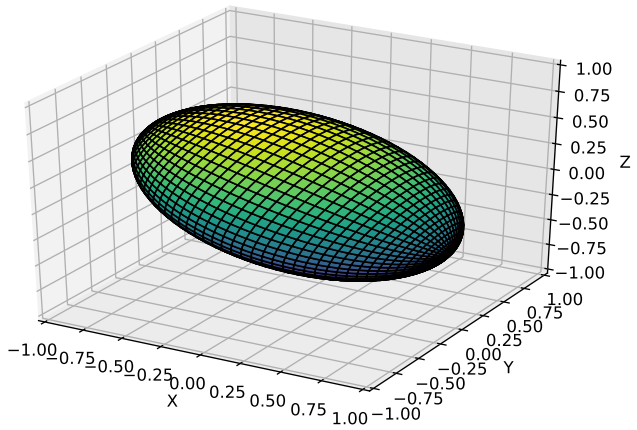


Figure 5.2: Equiprobability Surface of a Measurements Covariance

height for that particular point together with a height variance representing the certainty of the map thus can be retrieved.

This treats measurements with their measurement variance as a globally evaluatable scalar field. Naturally, the probability of the measurement being at one particular point in space is the highest at the measurement mean itself. Ellipsoids in the shape of the measurement covariance are surrounding the mean and form equal-probability surfaces. This yields a setup, where every measurement can be evaluated for the probability of actually resulting from a terrain feature at any point in space. A depiction of one equiprobability surface from a stereo measurement is shown in Figure 5.2.

5.2.1 *Query*

Ultimately, the height query can be performed with simple Bayesian inference. Since the map is queried at a specific point and therefore only a one-dimensional Gaussian distribution is needed, the 3D multivariate distributions cannot be fused directly.

The multivariate probability density function is expressed by:

$$P(x) = \frac{1}{\sqrt{(2\pi)^d \cdot |\Sigma_m|}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma_m^{-1}(x-\mu)\right) \tag{5.1}$$

For getting fusible 1D distributions, this distribution must be sliced in two axes. This can best be imaged as finding the probability distribution along a vertical arrow at the coordinates p_q. Further the resulting distri-

bution is referred to as $P(z|x,y)_p$. The p stands for projected. $P(z|x,y)_p$ is not a normal distribution, since:

$$\int_{-\infty}^{\infty} P(z|x,y)_p dz \neq 1 \tag{5.2}$$

This is intuitively explained by the fact, that it is far more probable, that the measurement in question is not on that 1D slice, then that it is. The distribution along the height vector still represents the correct relative probabilities which is described by$P(z|x,y)_p$. $P(z|x,y)_p$ is a scaled normal distribution with a mean, variance and scaling factor derivable by the original 3D multivariate distribution (5.1) and the query point.

The parameters of the sliced distribution are:

$$\mu_p = \frac{\sigma_{xz}}{\sigma_{xx}} \cdot (p_{q,x} - \mu_x) + \frac{\sigma_{yz}}{\sigma_{yy}} \cdot (p_{q,y} - \mu_y) + \mu_z \tag{5.3}$$

$$\sigma_p^2 = \sigma_{zz} - \frac{\sigma_{xz}^2}{\sigma_{xx}} - \frac{\sigma_{yz}^2}{\sigma_{yy}} \tag{5.4}$$

Acquisition of the prescaler is done by elimination of the remaining scaling factor of the 1D PDF and subsequent multiplication with the maximum probability of $P(z|x,y)_p$. The regular 1D PDF is given by:

$$P(z) = \underbrace{\frac{1}{\sqrt{2\pi \cdot \sigma^2}}}_{\text{prescaler}} \exp\left(-\frac{1}{2}(z-\mu)^2\sigma^2\right) \tag{5.5}$$

The used scaling factor for a regular PDF therefore is:

$$s_p = P(\left[q_x, q_y, \mu_p\right]^T) \cdot \sqrt{2\pi \cdot \sigma_p^2} \tag{5.6}$$

The final distribution is then:

$$P(z|x,y)_p = s_p \cdot P(z, \mu_p, \sigma_p) \tag{5.7}$$

Figure 5.3 shows the resulting distribution. For comparison the raw probabilities directly sampled from the 3D multivariate distribution are plotted as well.

Querying the map at a specific point is done by a simple weighted mean over all $P(z|x,y)_p$ with mean and variance.

5.2.2 *Pose Based Degradation*

Pose uncertainty of the rover directly affects the uncertainty of previous measurements. Since this map does not use bucketing or discretization,

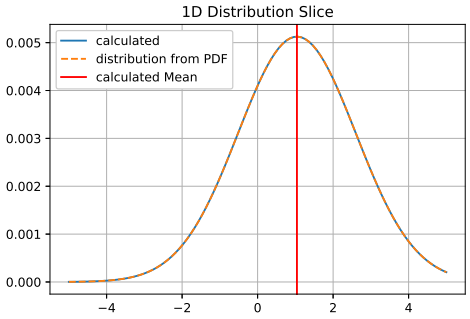


Figure 5.3: Sliced Distribution

pose uncertainty of the rover needs to be projected and added to each measurement covariance individually. Effectively, the shape and magnitude of the equiprobability surfaces change with added position and orientation uncertainty over time.

Each measurement uncertainty is updated by the following rule:

$$\Sigma_{m,t+1} = \Sigma_{m,t} + J_r \Sigma_r J_r^T + J_\Phi \Sigma_\Phi J_\Phi^T \tag{5.8}$$

With Σ_r and J_r being the covariance and Jacobi matrix of position uncertainty and Σ_Φ and J_Φ being the covariance and Jacobian of the rotational uncertainty. It is noteworthy, that J_Φ is strongly dependent on the distance and direction of the robot to the measurement in question.

This straight forward approach works, since the measurement variance is strictly decoupled from the sample variance. With uncertainty in the pose, the measurement precision of previous measurements degrades, but the sample variance of bucketed measurements can remain unchanged.

Since this map requires storage of each measurement with an attached covariance matrix forever, it quickly becomes unfeasible. The concept of considering multiple surrounding measurements for extrapolating the terrain can be implemented more efficiently.

5.3 CCM SURFACE REPRESENTATION

A hard requirement for all considered map updates is the ability to run incrementally. With each new measurement, an existing representation should be updated without the need to recompute the entire map or increase memory allocation. For those reasons, the previously presented optimal map is not feasible in practice and therefore replaced

by a discretized stochastic representation. Map cells must use constant memory and have to be queryable at all times.

In the most simple setup, a piecewise linear map requires three values per cell (*height*, *inclination X*, *inclination Y*). In reality, this representation lacks the ability to

- perform an incremental update
- use a different weight for individual measurements
- maintain a roughness estimate for each cell

To support these features a covariance based representation is introduced. Incrementally, a multivariate Gaussian distribution is fitted to the measurements. The multivariate Gaussian distribution can easily be interpreted as a two-dimensional least-squares plane fit. Hence, sub-cell accuracy height queries can be performed on this piecewise linear map. Ultimately, the true sample (co)variance [2] distribution is needed.

Mean and covariance can be updated incrementally with only one additional parameter representing the accumulated weight as derived in Section 5.4. Furthermore, inclination in X / Y direction and roughness with respect to the surface normal are easily recoverable as shown in Section 5.6.

Each cell contains the values outline in Table 5.1. In practice, all variance / covariance values are stored scaled by the accumulated weight in order to reduce computational complexity during the measurement update. Not all six unique elements of the covariance matrix are needed for update and query. The XY covariance is not expressive regarding the measurement distribution and thus not included.

5.3.1 *3-Dimensional Mean*

Due to a possibly unequal distribution of points in the lateral directions, the mean has to be estimated in all three dimensions. As illustrated in Figure 5.4 a lateral bias changes the estimated slope significantly.

5.3.2 *Sample Covariance*

Inclination and Roughness can be retrieved from the sample (co)variance values. For visualization a simple two-dimensional example is considered. The ground truth is a simple slope. Eight measurements are generated with simulated covariance based on stereo error. (The simulated camera is located at $x = -3; y = 0.5$)

2 In contrast to the measurement covariance which influences the sample covariance but has no informative value for reconstruction of the terrain.

CCM Elements		
ID	Function	Notation
0	accumulated weight	W
1	mean x	μ_x
2	mean y	μ_y
3	mean z	μ_z
4	Variance x	σ^2_{xx}
5	Variance y	σ^2_{yy}
6	Variance z	σ^2_{zz}
7	Covariance xz	σ^2_{xz}
8	Covariance yz	σ^2_{yz}

Table 5.1: CCM Cell Elements

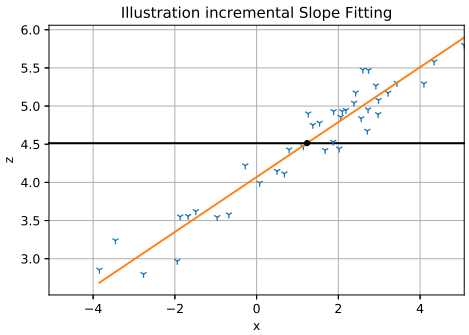


Figure 5.4: Non Center Distribution Resulting in X Offset.

The resulting measurements and covariance ellipses are shown in Figure 5.5. By definition, the 3σ ellipse includes 99.7% of all measurements. Therefore, its shape and orientation is dependent on the underlying observed surface. A simple first order approximation and surface roughness estimation can be retrieved. Figure 5.6 shows the same setup with a sub-cell sized obstacle. The sample variance in the direction normal to the estimated first order slope is greatly increased. This is visualized by the more bulky 3σ ellipse. Obstacle detection based on this projected sample variance is therefore decoupled from the surface inclination.

It is possible to counteract the effect of the added measurement covariance as described in Section 5.11. Due to the described drawbacks of the method, it is not applied in the final version of CCM and should be considered as an optional addition.

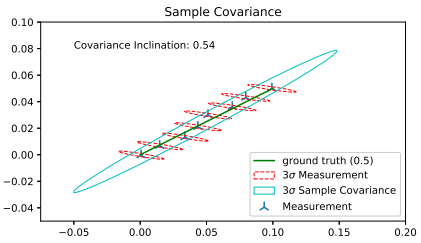


Figure 5.5: 2D Measurement Vector with Measurement- and Sample Covariance

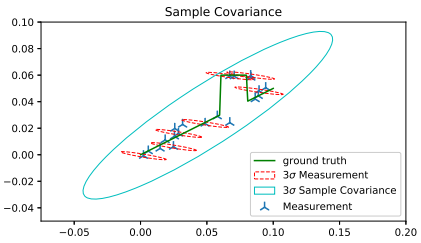


Figure 5.6: 2D Covariance Fit with Sub-Cell Sized Obstacle in Cell

5.4 COVARIANCE UPDATE

This section discusses the measurement update itself. It is the heart of the algorithm and gets executed every time a new measurement is added to the cell. To perform the incremental variance update in a numerically stable manner, σ^2 is not stored directly. Instead, a by the accumulated weight w scaled value is introduced for representing variance and further referred to as S.

In a more advanced version of CCM one weight parameter per axis can be used. This would allow for individual axis updates. With individual axis updates, one could take advantage of scenarios, where the same cell is observed from multiple points of view. The characteristically stretched covariance ellipse from stereo would hereby be used to weight measurements in the more precise dimension stronger. An overall improved accuracy is expected.

5.4.1 *Measurement Weighting*

Weighting of the measurements can be done in multiple ways. The essence of weighting is to represent the relative sensor precision of each measurement. This way, more precise (closer) measurements contribute stronger to the result than imprecise ones from a greater distance.

In case of a stereo camera, the measurement variance is dependent on the distance to the target 2.4. The weight can for example be calculated from the Measurement Covariance Matrix $\Sigma_{m,t}$ by using the determinant as in Equation (5.9) or with simple addition as in Equation (5.10). σ_{xy}^2 can safely be assumed as zero.

$$w_t = \frac{1}{|\det \Sigma_{m,t}|} \tag{5.9}$$

$$w_t = \frac{1}{\sigma_{xx}^2 + \sigma_{yy}^2 + \sigma_{zz}^2 + \sigma_{xy}^2 + \sigma_{yz}^2 + \sigma_{xz}^2} \tag{5.10}$$

This adaptive weighting is not fundamentally necessary for a successful application of CCM. Due to the high density of points present close to the sensor, a natural weighting by occurrence exists between the discretized cells. Cells close to the sensor update much faster, then distant cells simply due to more measurements associated with the cell. The final applied version of CCM uses a constant weight of 1 for each measurement, regardless of distance. To control the impact of measurements and the convergence rate, parameters described in Section 5.8 can be adapted. A well tuned convergence cap ensures fast adaptability to new and precise measurements.

5.4.2 *Incremental Update Equations*

The accumulated weight is updated by simple addition as in Equation 5.11. w_t is the measurement weight and W_t the accumulated cell weight. The subscript $_c$ marks cell associated values while $_m$ stands for measurement values. Post update values are indicted by $_t$ while old values are marked with $_{t-1}$.

$$W_t = W_{t-1} + w_t \tag{5.11}$$

For updating all following parameters, the measurement must first be transformed into a cell-centric frame by subtracting the cells' X- and Y center coordinates (C_x, C_y) from the measurement:

$$\mu_{m,t} = [M_x - C_x, M_y - C_y, M_z]^T \tag{5.12}$$

Subsequently, the cells' means $\mu_{c,t,[x,y,z]}$ can be updated incrementally:

$$\mu_{c,t,[x,y,z]} = \mu_{c,t-1,[x,y,z]} + \frac{w_t}{W_t} \cdot (\mu_{m,t,[x,y,z]} - \mu_{c,t-1,[x,y,z]}) \tag{5.13}$$

Furthermore, the variance associated values S_{xx}, S_{yy}, S_{zz} are updated:

$$\begin{aligned} S_{[xx,yy,zz],t} =& S_{[xx,yy,zz],t-1} + w_t \cdot (\mu_{m,t,[x,y,z]} - \mu_{c,t-1,[x,y,z]}) \\ & \cdot (\mu_{m,t,[x,y,z]} - \mu_{c,t,[x,y,z]}) \end{aligned} \tag{5.14}$$

Finally, the three covariance associated values S_{xy}, S_{xz}, S_{yz} are updated:

$$\begin{aligned} S_{[xy,xz,yz],t} =& S_{[xy,xz,yz],t-1} \\ & + w_t \cdot (\mu_{m,t,[x,x,y]} \cdot \mu_{m,t,[y,z,z]} - \mu_{c,t-1,[x,x,y]} \cdot \mu_{c,t-1,[y,z,z]}) \\ & + W_t \cdot (-\mu_{c,t,[x,x,y]} \cdot \mu_{c,t,[y,z,z]} + \mu_{c,t-1,[x,x,y]} \cdot \mu_{c,t-1,[y,z,z]}) \end{aligned} \tag{5.15}$$

All variances and covariance can be acquired by:

$$\Sigma_c = \begin{bmatrix} S_{xx}/W_t & S_{xy}/W_t & S_{xz}/W_t \\ S_{xy}/W_t & S_{yy}/W_t & S_{yz}/W_t \\ S_{xz}/W_t & S_{yz}/W_t & S_{zz}/W_t \end{bmatrix} \tag{5.16}$$

The derivation of Equation (5.13), (5.14) and (5.15) are shown in Section 5.4.3.

5.4.3 *Derivation of Incremental Updates*

The derivations of weighted incremental mean and variance are shown in [82]. I could not find a similar source for the weighted covariance. As a starting point, the incremental weighted mean update is presented first.

$$\mu = \frac{\sum_{i=1}^{n} w_i \cdot x_i}{\sum_{i=1}^{n} w_i} \tag{5.17}$$

$$W_n = \sum_{i=1}^{n} w_i \tag{5.18}$$

$$\mu_n = \frac{1}{W_n} \sum_{i=1}^{n} w_i x_i \tag{5.19}$$

$$= \frac{1}{W_n} \left(w_n x_n + \sum_{i=1}^{n-1} w_i x_i \right) \tag{5.20}$$

$$= \frac{1}{W_n} (w_n x_n + W_{n-1} \mu_{n-1}) \tag{5.21}$$

$$= \frac{1}{W_n} (w_n x_n + (W_n - w_n) \mu_{n-1}) \tag{5.22}$$

$$= \frac{1}{W_n} (W_n \mu_{n-1} + w_n x_n - w_n \mu_{n-1}) \tag{5.23}$$

$$= \mu_{n-1} + \frac{w_n}{W_n} (x_n - \mu_{n-1}) \tag{5.24}$$

Moving on, the weighted variance:

$$\sigma^2 = \frac{1}{W_n}\sum_{i=1}^{n} w_i(x_i - \mu)^2 \tag{5.25}$$

$$= \frac{1}{W_n}\sum_{i=1}^{n} w_i x_i^2 - \mu^2 \tag{5.26}$$

$$\text{Let } S_n = W_n\sigma_n^2 \tag{5.27}$$

$$= \sum_{i=1}^{n} w_i x_i^2 - W_n\mu_n^2 \tag{5.28}$$

$$S_n - S_{n-1} = \sum_{i=1}^{n} w_i x_i^2 - W_n\mu_n^2 - \sum_{i=1}^{n-1} w_i x_i^2 + W_{n-1}\mu_{n-1}^2 \tag{5.29}$$

$$= w_n x_n^2 - W_n\mu_n^2 + W_{n-1}\mu_{n-1}^2 \tag{5.30}$$

$$= w_n x_n^2 - W_n\mu_n^2 + (W_n - w_n)\mu_{n-1}^2 \tag{5.31}$$

$$= w_n\left(x_n^2 - \mu_{n-1}^2\right) + W_n(\mu_{n-1}^2 - \mu_n^2) \tag{5.32}$$

$$= w_n\left(x_n^2 - \mu_{n-1}^2\right) + W_n(\mu_{n-1} - \mu_n)(\mu_{n-1} + \mu_n) \tag{5.33}$$

$$= w_n\left(x_n^2 - \mu_{n-1}^2 + (\mu_{n-1} - x_n)\left(\mu_{n-1} + \mu_n\right)\right) \tag{5.34}$$

$$= w_n(x_n - \mu_{n-1})(x_n - \mu_n) \tag{5.35}$$

$$S_n = S_{n-1} + w_n\left(x_n - \mu_{n-1}\right)(x_n - \mu_n) \tag{5.36}$$

$$\sigma^2 = S_n/W_n \tag{5.37}$$

Finally, the weighted covariance:

$$\sigma_{xy} = \frac{1}{W_n}\sum_{i=1}^{n} w_i(x_i - \mu_x)(y_i - \mu_y) \tag{5.38}$$

$$= \frac{1}{W_n}\sum_{i=1}^{n} w_i(x_i y_i - x_i\mu_y - y_i\mu_x + \mu_x\mu_y) \tag{5.39}$$

$$\text{Let } S_{xy,n} = W_n\sigma^2_{xy,n} \tag{5.40}$$

$$= \sum_{i=1}^{n}(w_i x_i y_i) - W_n\mu_{x,n}\mu_{y,n} \tag{5.41}$$

$$S_n - S_{n-1} = \sum_{i=1}^{n}(w_i x_i y_i) - W_n\mu_{x,n}\mu_{y,n} - \sum_{i=1}^{n-1}(w_i x_i y_i) - W_{n-1}\mu_{x,n-1}\mu_{y,n-1} \tag{5.42}$$

$$= w_n x_n y_n - W_n\mu_{x,n}\mu_{y,n} + W_{n-1}\mu_{x,n-1}\mu_{y,n-1} \tag{5.43}$$

$$= w_n x_n y_n - W_n\mu_{x,n}\mu_{y,n} + (W_n - w_n)\mu_{x,n-1}\mu_{y,n-1} \tag{5.44}$$

$$= w_n(x_n y_n - \mu_{x,n-1}\mu_{y,n-1}) + W_n(-\mu_{x,n}\mu_{y,n} + \mu_{x,n-1}\mu_{y,n-1}) \tag{5.45}$$

$$S_n = S_{n-1} + w_n(x_n y_n - \mu_{x,n-1}\mu_{y,n-1}) + W_n(-\mu_{x,n}\mu_{y,n} + \mu_{x,n-1}\mu_{y,n-1}) \tag{5.46}$$

$$\sigma_{xy,n} = S_n / W_n \tag{5.47}$$

5.5 POOLING

This Section describes the pooling process for CCM. Only pooling from fine to corse layers is considered as this supports the natural flow of information. A new measurement is only inserted into the highest available resolution during measurement update. Upon query of the map, pooling is executed, and all acquired information flows from the high resolution layers into the low resolution ones.

While this is a trivial weighted average for the cell means ($\left[\mu_x, \mu_y, \mu_z\right]$) the covariance merge needs more attention. The accumulated cell weight W_{t-1} is used as a weighting factor.

5.5.1 *Covariance Combination*

Since the high-resolution cells which shall be combined do not observe the same terrain patch, conventional Gaussian inference cannot be applied for this operation. Means of the cells are not different due to measurement noise, but by design (neighboring cells). The covariance combination aims to provide the same result, as if every point had been inserted into the corse layer in the first place.

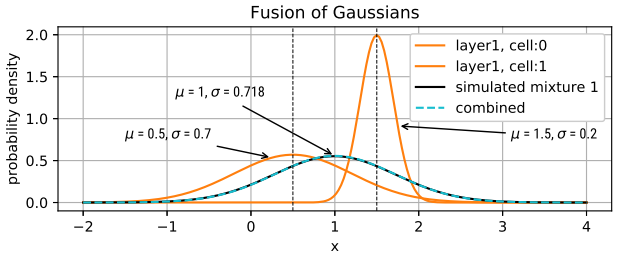


Figure 5.7: Variance Combination Example

Instead of performing conventional mixture of Gaussians, both distributions need to be transformed to the new mean. The new combined mean is simply the mean of means. For the variance, the projection is handled in an additive term as shown in Equation (5.49). Capital letters refer to the corse resolution cell while small letters describe one of the sub-cells in the higher resolution layer.

$$\mu_C = \frac{\mu_{c,1} + \mu_{c,2}}{2} \tag{5.48}$$

$$\sigma_C^2 = (\mu_{c,1} - \mu_C) \cdot (\mu_{c,2} - \mu_C) + \frac{\sigma_{c,1}^2 + \sigma_{c,2}^2}{2} \tag{5.49}$$

The combination update is also visualized in Figure 5.7.

This merge of Gaussians can easily be extended to 2.5D covariances.

$$\mu_C = \frac{\Sigma_{i=0}^4 \mu_{c,i}}{4} \tag{5.50}$$

$$\sigma_{C,[x,y,z]}^2 = \frac{\Sigma_{i=0}^4 \sigma_{c,xyz,i}^2 + (\mu_{C,xyz} - \mu_{C,i,xyz})^2}{4} \tag{5.51}$$

$$\sigma_{C,[xz,yz]}^2 = \frac{\Sigma_{i=0}^4 \sigma_{c,xzyz,i}^2 + (\mu_{C,xy} - \mu_{C,i,xy}) * (\mu_{C,z} - \mu_{C,i,z})}{4} \tag{5.52}$$

5.5.1.1 *Derivation Covariance Combination*

Observation of a Gaussian distribution from an offset mean is derived further.

$$\sigma^2 = E[(x-\mu)^2] \tag{5.53}$$

when the center of observation μ is shifted by a:

$$\sigma_a^2 = E[(x-\mu+a)^2] \tag{5.54}$$

$$\sigma_a^2 = E[a^2 + 2ax - 2a\mu + \underbrace{x^2 - 2x\mu + \mu^2}_{(x-\mu)^2}] \tag{5.55}$$

$$\sigma_a^2 = E[a^2 + 2ax - 2a\mu] + \sigma^2 \tag{5.56}$$

$$\sigma_a^2 = a^2 - 2a\mu + 2a \cdot E[x] + \sigma^2 \tag{5.57}$$

$$\sigma_a^2 = a^2 - 2a\mu + 2a\mu + \sigma^2 \tag{5.58}$$

$$\sigma_a^2 = a^2 + \sigma^2 \tag{5.59}$$

5.5.2 *How to Handle Existing Measurements in Low Resolution Layers*

The previously described pooling operation only combines the high-resolution cells to one cell of combined size. This can be used directly for the low resolution layer if no previous information was present on that layer. Otherwise, each subsequent pooling would introduce a bias towards the combined cells, underestimating the weight of previous measurements in the low resolution layer.

This issue can be taken care of by storing a copy of all layers but the highest resolution. As soon as a cell on any of the low resolutions enters the area which is covered by a high resolution layer, its value gets copied to the auxiliary map. The auxiliary map is only used as the low resolution prior for the pooling, but does not get overwritten by it. Pooling results are still stored in the original data structure.

By following this scheme, it is ensured, that old values are not overwritten by the pooling operation, preventing a *convergence* to the pooled value. The copy concept was previously described by [57]. In some applications it might be sound to omit this nuance and just overwrite existing values by the combined high resolution data. Usually, combined high resolution data has a higher precision than old measurements from a high distance. It remains to be noted, that this operation is necessary for a *correct* representation.

5.6 INCLINATION CALCULATION

To calculate the cell inclination i in X and Y direction (i_x, i_y) Equation (5.60) is evaluated.

$$i_{[x,y]} = \frac{\sigma^2_{[x,y]z}}{\sigma^2_{[x,y]}} \tag{5.60}$$

Proof:

$$\begin{aligned} i_x &= \frac{\sigma^2_{x,z}}{\sigma^2_{x,x}} \\ \sigma^2_{x,z} &= \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N-1} \\ \sigma^2_{x,x} &= \frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1} \\ i &= \frac{\sum_{i=1}^{N}(x_i - \bar{x})(z_i - \bar{z})}{\sum_{i=1}^{N}(x_i - \bar{x})^2} \\ i &= \frac{\sum_{i=1}^{N}(z_i - \bar{z})}{\sum_{i=1}^{N}(x_i - \bar{x})} \end{aligned} \tag{5.61}$$

Typically, only the magnitude of the total inclination is of interest for hazard segmentation. Πυθαγορας ο Σαμιος [3] has come up with a handy equation for this issue:

$$i_{total} = \sqrt{i_x^2 + i_y^2} \tag{5.62}$$

5.7 HEIGHT AND VARIANCE QUERY

Due to the piecewise linear representation of the surface, map queries with sub-pixel accuracy can be performed. Both, mean height and projected variance can be queried. The height $\mu_{c,z}$ and variance σ^2_{zz} at a location $q = [q_x, q_y]^T$ are requested.

First, the query point is matched to the containing cell and the coordinates expressed in cell-centric coordinates.

$$q_c = [q_x - C_x, q_y - C_y]^T \tag{5.63}$$

3 Also known for his remarkable work on right angles.

The sub-cell height h_q is then calculated by:

$$a = \frac{\sigma_{xz}^2}{\sigma_{xx}^2} \tag{5.64}$$

$$b = \frac{\sigma_{yz}^2}{\sigma_{yy}^2} \tag{5.65}$$

$$c = \mu_{c,z} - a \cdot \mu_{c,x} - b \cdot \mu_{c,y} \tag{5.66}$$

$$h_q = c + a \cdot q_{c,x} + b \cdot q_{c,y} \tag{5.67}$$

Directly returning σ_{zz} for the variance would represent the total height variance of that cell. This variance includes the slope induced spread and is therefore of limited usefulness. Piecewise linear maps with a correct variance representation would estimate the same value.

To get the true roughness without slope induced sample variance a conditional variance needs to be calculated. Similar to 5.2 the variance can be expressed as a sliced covariance ellipsoid. The inclination projected roughness is calculated by (5.68). σ^4 is the squared variance.

$$\sigma^2 = \sigma_{zz}^2 - \frac{\sigma_{xz}^4}{\sigma_{xx}^2} - \frac{\sigma_{yz}^4}{\sigma_{yy}^2} \tag{5.68}$$

5.8 INFINITE INTEGRATION SUPPRESSION

The incremental update of mean and variances relies on the calculation of multiple infinite sums. Particularly the summation of large variance values can quickly exceed the limitations of single precision floating point numbers. A mechanism is introduced to limit the endless integration. This is achieved by setting an upper weight limit. When a new insertion supersedes the weight limit W_l, the difference to the limiter is used to restrain all values. The following equations are only applied when the cell weight exceeds the threshold:

$$\psi = W_t / W_l \tag{5.69}$$

$$W_{t+1} = W_t / \psi \tag{5.70}$$

$$S_{[xx,yy,zz,xz,yz]t+1} = S_{[xx,yy,zz,xz,yz],t} / \psi \tag{5.71}$$

As a surplus, the limiter weight controls the minimal convergance rate. Once it is reached, each new measurement has the same relative impact on the values. Over time, a fully converged cell [4] can still adapt to changes in the measurements. This factor increases fault tolerance and

4 As in the weight limiter is reached.

error resilience. Selecting a lower weight limit results in less confidence in the converged cell regarding new measurements and therefore fast adaption.

5.9 HAZARD DETECTIEON

CCM's hazard detection is split into variance and slope based classification. The projected variance estimates the presence of sub-cell sized obstacles. Slope estimation is directly used for traversability analysis of inclined areas. Large obstacles, which are well described as first order slopes at the local extent of high resolution cells also are detected by this threshold.

Both techniques require, that evaluated cells are well and uniformly covered with measurements to avoid false classification of partially observed cells. Coverage is estimated by comparing the lateral measurement variances with the expected variance of a uniform distribution of cell size. Cells with coverages greater than 80% are evaluated for hazards. Other cells are classified as unknown.

5.9.1 *Variance*

In contrast to other mapping approaches CCM implicitly includes measures for inclination and roughness. Those figures can easily be used for in-place hazard detection. To detect sub cell sized obstacles, a simple variance threshold is applied. The variance query from Section 5.7 represents the sample deviation from the fitted plane. Threshold values are dependent on expected obstacle shape, size and minimal coverage of the cell. This value is estimated with a simple simulation as later demonstrated in Section 7.3.2. Selection of this value is highly application and sensor specific.

5.9.2 *Inclination*

Slope obstacles need to be distinguished by the level resolution. High resolution layers are capable of resolving slopes, which are significantly shorter than the rover itself. This can result in setups, where the local inclination of one cell is higher than the traversable maximum, but the true inclination of the rover will be within limits due to a significantly larger wheelbase. This situation is illustrated in Figure 5.8. Even though the area would be safe to traverse, path planning algorithms will avoid it. The inclination of individual high resolution [5] cells can therefore only be used to detect rock obstacles. Obstacles, which by themselves are slope shaped (are well approximated by a first order fit) are not

5 Significantly smaller than the rover wheelbase

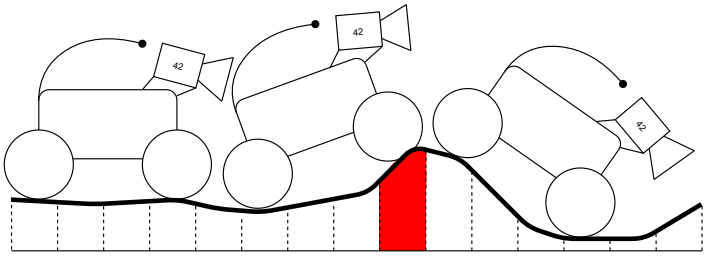


Figure 5.8: Small Inclination Patches are not Representative for Driveability.

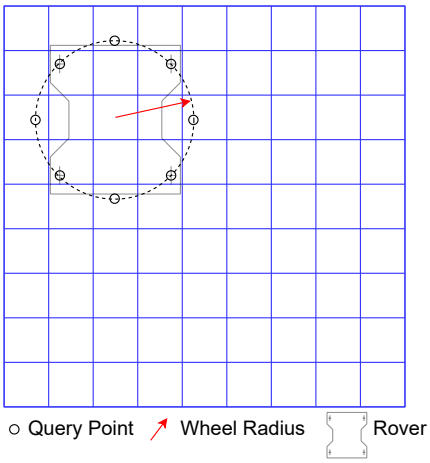


Figure 5.9: Inclination Query for High Resolution Levels.

detected by the variance based detection. The inclination threshold of high resolution cells therefore must be set to detect the minimal obstacle height expressed as a slope.

For a 4 cm cell and 5 cm minimal obstacle size, this implies a slope threshold of 51.3°.

To detect *true* slope induced traversability hazards on high resolution layers, a second evaluation is needed: For each cell center, eight heights on the wheelbase perimeter are sampled. A plane can be fitted to those height points for reconstructing the estimated vehicle inclination. Figure 5.9 shows the sample points. Due to the piecewise linear nature of CCM the actual sub-cell height can be used for this sampling. Finally, the estimated vehicle inclination can be thresholded with the vehicle's performance parameter.

This vehicle size consideration is only necessary for cells which are smaller than the wheelbase. Larger cells implicitly estimate the correct slope and are directly used for traversability classification.

5.10 CELL SIZE AND MEASUREMENT COUNT PARAMETERS

Similar to other mapping approaches, certain parameters of the map need to be adapted to meet the requirements of the entire system while obeying constraints induced by the sensor and localization. Among these adaptable parameters are

- Cell Size
- Layer Stackup (How Far Each Cell Size Reaches)
- Variance Threshold for Obstacle Detection
- Convergence Weight Limiter

These parameters are constraint by measurement variance and measurement count per cell.

5.10.1 *Variance Constraint*

CCM is particularly sensitive to increasing measurement variance and low measurement count. Correct choice of map resolution is more relevant compared to classic piecewise constant approaches.

To ensure proper slope estimation, the resulting sample covariance ellipsoid must have two distinct major axes and one significantly smaller one. Due to the increasing measurement variance with distance, there is a threshold from which on the cell size is too small to ensure proper convergence. This effect is illustrated in Figure 5.10. The shown example is generated by simulating a single cell with 100 samples. Samples are degraded based on simulated stereo noise. The movement direction is along the X-axis.

At approximately 15 m distance, the map loses the associated inclination. In Y-direction no such effect is visible. This can be explained by asymmetric nature of the measurement variances. Since the distance to the cell is only increased in X-direction, the measurement covariance is also aligned with this axis. Even at 50 m, the measurement standard deviation in Y-direction has not reached the tipping threshold.

To calculate the cutoff point, only measurement variances and cell sizes need to be considered. Satisfying the necessary requirement of having two distinct major axes in the sample covariance can only be achieved as long as the measurement variance is lower than the expected variance of the uniform distribution (5.72) along the lateral directions. This is intuitively explained with the idea, that lateral variance needs to result from the width of the cell rather than uncertainty in the measurement

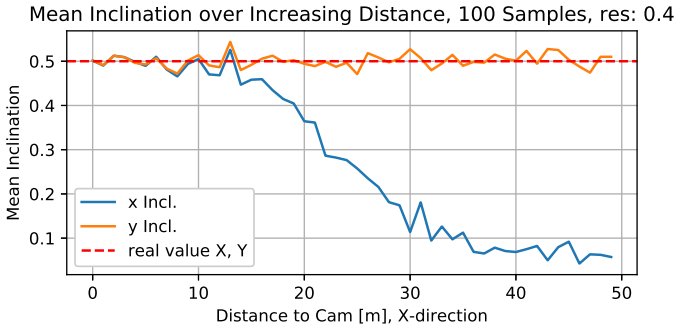


Figure 5.10: Inclination Convergence over Distance

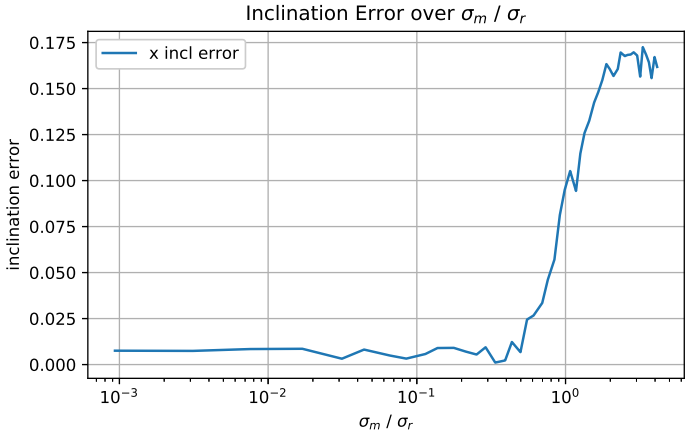


Figure 5.11: Inclination Convergence over Fractional Standard Deviation

to correctly sort points within the cell. Only if this is given, a proper slope estimation is feasable.

$$\sigma_r = \sqrt{\frac{w^2}{12}} \tag{5.72}$$

As soon as the measurement standard deviation in x/y direction has the same magnitude as the maximum expected standard deviation from cell coverage, inclination estimation becomes unfeasible and the error increases. This is illustrated in Figure 5.11, where the inclination error is plotted against the fraction of measurement standard deviation and maximum cell standard deviation from a uniform distribution of cell size.

By constraining this ratio to e.g. 0.5 and evaluating a setup based on stereo depth, range plots such as Figure 5.12 can be used for parameter selection. The blue covered area is representing valid cell size choices.

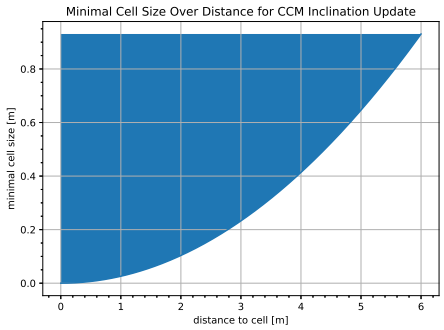


Figure 5.12: Minimal Cell Size for Given Distance

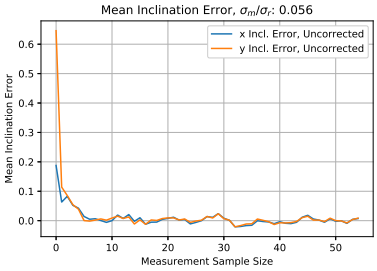


Figure 5.13: Inclination Convergence High Fraction

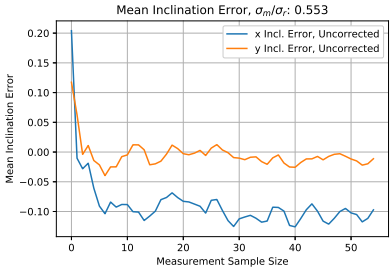


Figure 5.14: Inclination Convergence Low Fraction

5.10.2 *Measurement Count Constraint*

The second important parameter is how many measurements per cell are required for convergence. Figure 5.13 shows an exemplary inclination convergence as a function of samples per cell. The convergence rate and quality is partially dependent on the previously described standard deviation fraction and therefore observation distance. Since it is only feasible to push the ratio to about 0.6, the convergence rate at this margin is of interest. Figure 5.14 shows a convergence example right at the tipping point. It is already visible, that the cell does not converge to the right value due to the increasing offset caused by the suboptimal standard deviation ratio. For the inclination, a convergence at about 5 samples is visible.

The same experiments were conducted with a variety of ground-truth shapes. Fortunately, no shape dependency was observed.

A similar analysis can be performed for the variance convergence. Ground truth variances in height and plane divergence are compared to $\sigma^2_{c,zz}$ and σ^2_q. σ^2_q is calculated by:

$$\sigma_q = \frac{\sum_{i=0}^{n} (p_{i,z} - \mu_z - inc_x \cdot p_{i,x} - inc_y \cdot p_{i,y})^2}{n-1} \tag{5.73}$$

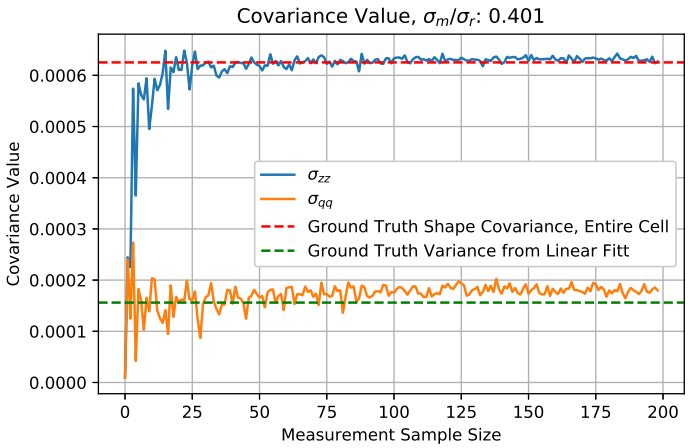


Figure 5.15: Variance Output Convergence

Visible in Figure 5.15 the variance estimation σ_q^2 slightly overestimates the ground truth plane divergance variance. This is easily explained by the super positioned measurement variance which changes the perceived inclination. The simple z variance on the other hand fits well.

Variance convergence itself shows the same independence on surface features as with the slope estimation. A minimum sample size of 10-20 is a good starting point for a reliable variance value.

5.11 OPTIONAL: MEASUREMENT VARIANCE COMPENSATION

Since the rotation of the measurement covariance ellipse does not describe the terrain, its impact on the sample variance can be subtracted from the final result. The corrected covariance describes the true terrain induced variance more precisely and removes any bias on the resulting inclination.

When the measurement count is high enough, the measurement sample covariance is a superposition of the measurement variance and the *true* variance caused by terrain features. Subtracting the mean measurement covariance can compensate for this effect and makes the variance caused by features in the terrain visible. This also eliminates the bias in rotation as shown in Figure 5.16.

The rotation effect is strongly dependent on the incidence angle from the camera to the cell. In the following example, the disturbing variance has a higher impact on the fitted sample covariance ellipse. Similarly to the example above, subtraction of the measurement variance eliminates the induced rotation. It is illustrated in Figure 5.17. This technique enhances the variance estimation, if the underlying surface causes a variance itself as depicted in Figure 5.18.

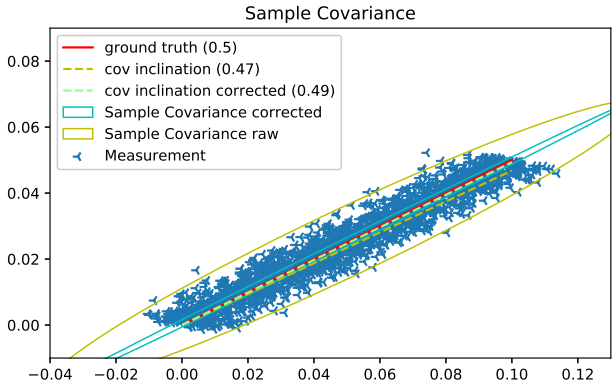


Figure 5.16: Sample Covariance Correction with Measurement Covariance

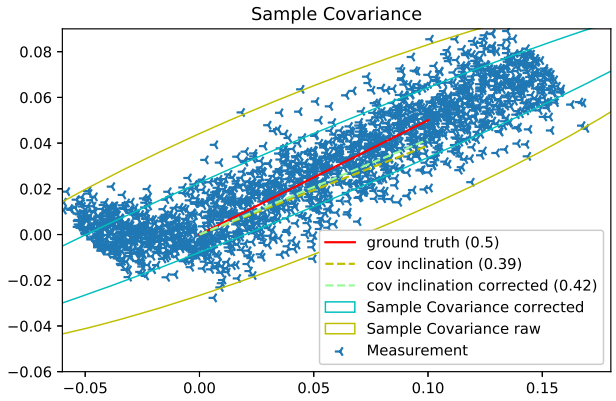


Figure 5.17: Sample Covariance Correction with Measurement Covariance From High Inclination

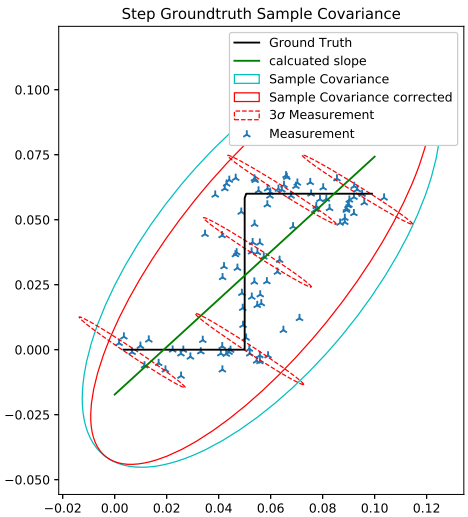


Figure 5.18: Step Ground-Truth with Covariance

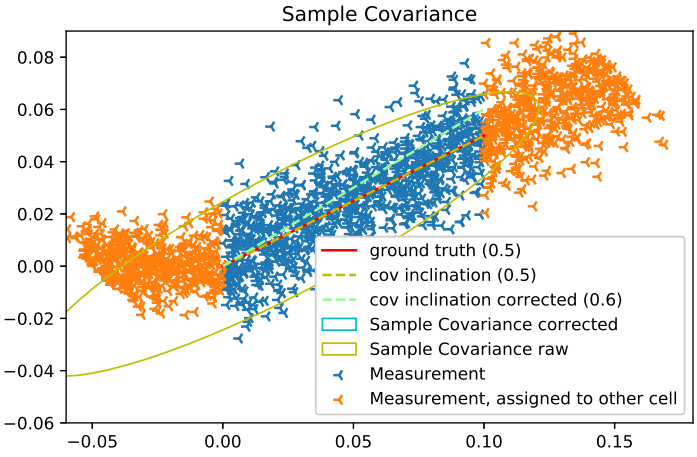


Figure 5.19: Cell Assignment difference

5.11.1 *Problems With Measurement Bias Correction*

The main difficulty with correcting the sample covariance by considering the mean measurement covariance lies in the discretizing nature of a grid-map. In particular measurements close to the border of each cell can be miss-associated due to measurement error. Each cell will therefore miss measurements and include measurements from neighboring cells.

The previous theoretical considerations have not included the discretization. Cutting measurements which would be associated to an adjacent cell already weakens the measurement variance impact significantly. Therefore, naively doing the correction will even degrade the inclination estimation in reality. This could be compensated for by re-considering measurements outside the cell in a band of 3σ, but this introduces even more difficulties. Keeping things simple: the bucketing nature of the map architecture pushes the solution into the right direction and partially compensates for the described measurement bias. The cell assignment and resulting inclinations are shown in Figure 5.19.

A second reason, why the measurement correction is disadvantageous becomes evident in the three-dimensional example. The following evaluation is performed on simulation data of a $10x10$cm cell at increasing distance and stereo-error. As ground truth, a simple inclined plane was used with a slope of 0.5 in both directions. Visible in Figure 5.20, at a certain distance, the error in inclination spikes for the corrected version. This happens when the measurement standard deviation in that particular axis is equal to the sample standard deviation. In this case, the inclination becomes undefined due to a division by zero. Figure 5.21 shows this dependency. After the spike in inclination error, the resulting

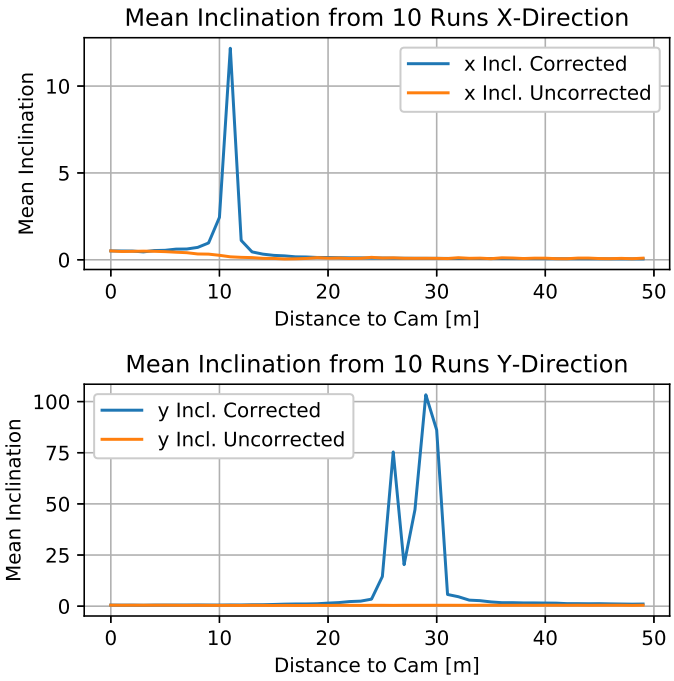


Figure 5.20: Correction Error over Distance to Cell

inclination is hijacked by the direction of measurement variance analog to the previously described maximum cell distance issue.

In conclusion, the measurement bias correction does not have significant real-world benefits and is therefore not considered further.

5.12 OPTIONAL: INDIVIDUAL AXIS UPDATE

The presented individual axes update is similarly optional as 5.11.

Some mission profiles might observe the same areas from different viewpoints. This could for example be achieved by utilizing multiple vehicles updating a common map or driving in certain patterns.

As mentioned before in 5.10.1 the measurement precision must not be equal for all map directions. Especially stereo cameras have a significantly higher measurement variance in the depth direction compared to the two normal directions. This can be exploited by introducing individual axis updates to the map.

Each cell gets extended to carry one additional accumulated weight parameter. By projecting the measurement covariance to the maps principal axis, the inverse variance can be used as an axis specific weight for that particular measurement. This way, the high precision direction (normal to the measurement vector) gets weighted much higher com-

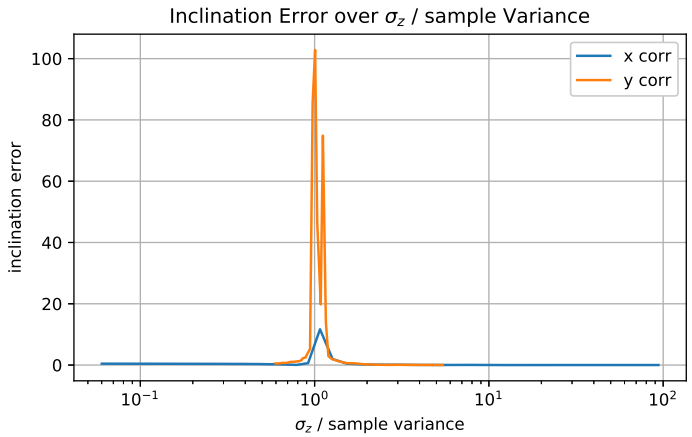


Figure 5.21: Inclination Error as a Function of Variance Quotient

pared to the low precision depth direction. Keeping track of individual weights for X- and Y direction in the map allows for scenarios, where the cell gets re-observed, and the current low-precision direction has been previously updated with high-precision measurements. New low precision measurements do not get accounted for as much as the high precision direction.

This concept has been explored in preliminary simulations, but does not get applied in the CADRE mission. Mission scenario, increased complexity and simple lack of necessity led to this decision.

6

MAP UPDATE ALGORITHMS

This chapter introduces different mapping techniques covering the space of 2.5D maps. A pre-selection of promising algorithms is made based on this high level comparison of temporal depth map fusion techniques. Even though many more update algorithms were experimentally tested at this evaluation stage, only the three most promising ones are presented in this thesis. All additional techniques performed either worse than one of the presented ones at same computational cost or were in general unfeasible. Kalman, OMG and CCM produced the most promising results.

6.1 INTRODUCTION

The application of mobile robotics on the lunar surface does not require a high fidelity 3D map representation as many other mobile robotics fields do. No overhangs, caves or arches are expected or need to be traversed. The main requirement for the presented mapping approaches is a reliable derivation of a binary [1] traversability map. Using geometric approaches, traversability is directly estimated from the terrain representation.

Full 3D maps are in general too computationally and memory intensive, do not translate well into a two-dimensional traversability map and are simply not required for the given task.

The presented mapping approaches are evaluated on a very simple simulated dataset.

The goal of this evaluation is to provide a relative overview of the performance of different mapping techniques. No determined value has absolute relevance but puts the different maps into perspective. Additionally, advantages and drawbacks of the approaches are visualized and discussed. Since a good geometric traversability analysis requires a precise map, approaches are evaluated based on their terrain accuracy.

We start with a reference map which does not have any practical usability in real world applications. Its sole purpose is to set a baseline and to improve understanding of the used metrics.

1 in practise we are dealing with a ternary map: [unknown, traversable, non-traversable]

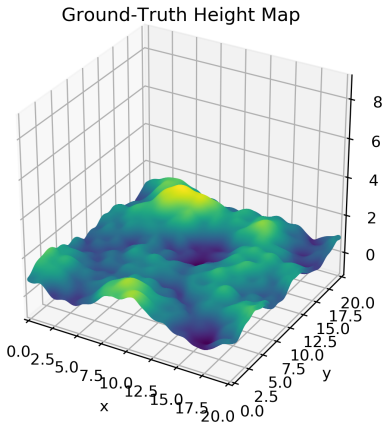


Figure 6.1: Demonstration Ground Truth DEM

6.2 DEMONSTRATION SETUP

At this stage of evaluation an accurate representation of the expected terrain is not inherently important. A simple procedurally generated height map is used as ground truth. Some parameters and therefore measurement assumptions will be altered during evaluation in order to demonstrate the maps' sensitivity for that particular parameter.

The procedurally generated height map used in all upcoming evaluations is shown in Figure 6.1.

To generate the measurement vector, this ground truth DEM is converted to a point cloud by random sampling. The samples are degraded afterwards by adding Gaussian noise to the point coordinates. In its simplest form, this noise is uniformly distributed in each axis. For more advanced analysis, the degrading covariance is calculated based on stereo noise. Position of the simulated camera among all other stereo influencing parameters are set according to the demonstration case.

In addition to point cloud degradation a with distance decreasing point density can be added.

To get an initial understanding of the different mapping techniques uniform noise and no density gradient is used for the evaluations.

6.3 EVALUATION EXAMPLE ON BASELINE BANANAS MAP

6.3.1 *Description of the Map Update*

Following the demonstrative intention of this mapping technique, its incremental update is rather simple. Each cell only stores one single height value. Measurements from a depth sensor gets treated as individual points of a point cloud. Prior to insertion, the point cloud is

transformed into the map frame. When a set of points gets added to the map, each point is associated with its corresponding map cell based on the x- and y coordinates of the point. Each map cell gets simply set to the height of the points' Z coordinate. Therefore, after the update each cell simply has the height value of the last added point.

6.3.2 *Obvious Drawbacks*

The most obvious drawback of this method is its non averaging nature. Whichever measurement comes last defines the value of a particular cell. This way any noise and deviation from the true value manifests in the final result. It is expected, that the *Bananas Map* shows the worst map terrain precision of all presented methods.

6.3.3 *Evaluation*

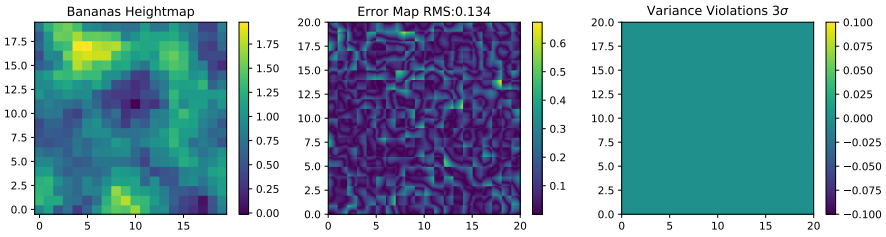


Figure 6.2: Bananas Map 1 m Cell Size Evaluation.

Figure 6.2 shows the evaluation plots for the 1 m resolution Bananas map. Besides the piecewise constant height map produced by the algorithm, an error map and variance violation plot is used for evaluation.

The error plot shows the absolute divergence of the maps' height estimate from the ground truth height. Additionally, the title indicates the RMS error between ground truth and map values. For this evaluation, the map is evaluated on ground truth resolution. Therefore, cell boundaries are clearly visible as discontinuities in the error field.

Unsurprisingly, each cell has a single point of small divergence. No obvious rule is visible where in the cell this low error point lies. This behavior aligns with our expectations of a map that just uses one single measurement for updating the height of a cell.

The last plot *Variance Violations* is not applicable for the Bananas Map. For maps, which incorporate some sort of variance estimation, this plot indicates areas where the ground truth does not lie within the 3σ band of the estimated height. This metric is useful to state whether the estimated variance can be used as a metric of roughness / awareness of the cell.

The Bananas Map unfortunately does not have a Variance estimate.

6.4 KALMAN

Using a simplified Kalman Filter measurement update in each cell estimating the cells' height is the idea behind the Kalman Map. The presented implementation is heavily inspired by [30]. Its simplicity and overall robustness makes the Kalman Map one of the Industries standard for mobile hazard avoidance maps. The Kalman Map models the point (height) distribution within one cell as a single Gaussian distribution.

6.4.1 *Description of the Map Update*

Each map cell consists of two values: μ_z (mean height) and σ^2_{zz} (Kalman height variance). Similar to the Bananas map from Section 6.3.1 depth sensor measurements are converted into a set of points and rotated to the map frame prior to insertion. Each point is treated as an individual measurement for the corresponding map cell. Map cell correspondences are based on x- and y coordinates only. A simple one dimensional Kalman filter measurement update is used to update $\mu_z$and σ^2_{zz}. (6.2) shows the used equations. σ^2_m represents the measurement variance in Z direction for that particular point. It is acquired with the sensor model and also rotated into the map frame. μ_m is the height value of the measurement.

$$\sigma^2_{zz,n+1} = \frac{\sigma^2_{zz,n} \cdot \sigma^2_m}{\sigma^2_{zz,n} + \sigma^2_m} \tag{6.1}$$

$$\mu_{z,n+1} = \frac{\sigma^2_m \mu_{z,n} + \sigma^2_{zz,n} \mu_m}{\sigma^2_{zz,n} + \sigma^2_m} \tag{6.2}$$

Querying this map is as simple as returning the mean and variance for any given cell. Depending on the use case, the Kalman map can be extended to incorporate pose drift or time based variance inflation. Those elements would be implemented as the Kalman System Update.

When adding pose uncertainty, the estimated variance becomes a 3×3 covariance matrix. Only for querying, this covariance matrix is converted to a single height variance. Based on the movement of the rover and the system model, cell variances can be updated as previously described in 2.5.

Since in this technique, the variance is solely dependent on the number of measurements added to a cell, the cell variance can become too low for new measurements actually having an impact. Adaptability

to a changing environment or to compensate for localization errors is limited.

The convergence issue from Kalman is solved by introducing a simple time inflation. Re-convergence to a changing environment can be handled in two ways:

1. time inflation
2. convergence cap

Time inflation is a simple addition to the estimated variance value each time the map gets updated. This way, unobserved regions loose map confidence and new measurements get an increased impact on the map values.

A convergence cap needs to be introduced additionally. In the case of a still rover, the same cells get observed many times. Consequently, the map confidence rises to a state, where the map essentially is unchangeable. Introducing a limiter on the variance value keeps the map compliant to a certain extent. Each new measurement after the limiter is reached has the same relative weight. By choosing this value, map adaptability after the *final* convergence can be controlled.

6.4.2 *Evaluation*

The Evaluation results for the Kalman Update are shown in Figure 6.3. On a first impression, the height map does not differ much from the previously presented Bananas map. This is easily explained by the very low levels of noise present in this evaluation run. All measurements resemble ground truth height well.

The error map on the other hand shows substantial differences. Each cell has its zero error region crossing the center of the cell. This is mainly caused by the uniform distribution of points in x and y direction, but shows the benefits of an averaging map over the naive approach. Usually the ground truth terrain within one map cell has approximately continuous slope and therefore centers the height mean also in X and Y. The overall RMS error is unsurprisingly significantly lower than for the Bananas map.

Inspecting the Variance Violations plot it becomes evident, that the Kalman Map does not represent the surface features as part of the height variance. Since the variance always reduces slightly with each added measurement it does not represent the sample variance in any way. The Kalman filter is designed to estimate singular values instead of measuring the properties of a distribution. This is underlined when looking at the variance plot. It does not contain any information about

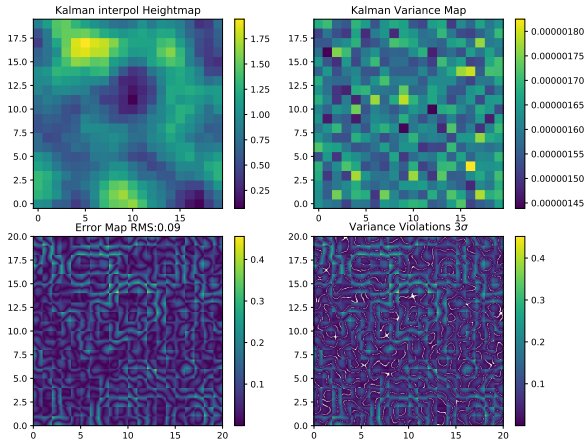


Figure 6.3: Kalman Map 1 m Cell Size Evaluation.

the underlying terrain and has absurdly small values. The variance can be seen as a *confidence* value, being a result of how many low variance measurements have already been added to the cell.

6.5 OMG

OMG is an advancement of the Kalman map. Most of the used technique is identical except that OMG actually estimates the sample variance in height. This is possible by introducing one additional *weight* parameter to each cell. Cell values are: μ_z, σ^2_{zz} and S_z. The OMG algorithm was first presented by [57].

6.5.1 *Description of the Map Update*

The estimated mean μ_z of OMG is identical to the Kalman approach.

An incremental estimation of mean, variance and S can be derived from the known mixture of Gaussian, which is described as:

$$f(x) = \frac{1}{S}\sum_{n=1}^{M} \sigma_{zz_i}^{-2} N(z_i, \sigma^2_{zz_i}) \tag{6.3}$$

$$S = \sum_{n=1}^{M} \sigma_{zz_i}^{-2} \tag{6.4}$$

This can be broken down to the mean and variance from it:

$$\mu = \frac{1}{S}\sum_{n=1}^{M} \frac{\mu_{m,i}}{\sigma^2_{zz_i}} \tag{6.5}$$

$$\sigma^2 = \frac{1}{S}\sum_{n=1}^{M} \frac{\sigma^2_{zz_i} + \mu^2_{m,i}}{\sigma^2_{zz_i}} - \mu^2 \tag{6.6}$$

These three equations can be re-formulated for an incremental update:

$$S_t = S_{t-1} + \sigma_{zzt}^{-2} \tag{6.7}$$

$$\mu_{z,t} = \frac{1}{S_t}\left(S_{t-1}\mu_{z,t-1} + \frac{\mu_{m,t}}{\sigma_{zzt}^2}\right) \tag{6.8}$$

$$\sigma_{zz,t}^2 = \frac{1}{S_t}\left(S_{t-1}(\sigma_{zz,t-1}^2 + \mu_{z,t-1}^2) + \frac{\mu_{m,t}^2}{\sigma_{zz,t}^2} + 1\right) - \mu_{z,t}^2 \tag{6.9}$$

Distinguishing between weight and variance enables correct estimation of the sample variance.

Map convergence can be steered in the same way as it is handled for the Kalman update. A time inflation and convergence limiter control the map adaptability to a changing environment or errors in pose estimation. For OMG, the S value needs to be adapted instead of the variance.

6.5.2 *Evaluation*

Figure 6.4 shows the analysis of the OMG map run. Height and thus Error map are identical to the previously analyzed Kalman map. It comes to no ones surprise, that the RMS error of 0.09 is also identical. OMG therefore has no precision advantage over Kalman.

The major difference becomes vivid when comparing the variance map and especially variance violations. Variance values are in a much more realistic range and show an increase in value at regions of high inclination. This is caused by the piecewise constant maps inability to approximate the inclined surface well. Naturally, more sample variance occurs and correctly shows up in the variance map.

By design, the variance violations are non-existent in this scenario. This demonstrates OMG's ability to estimate the sample variance correctly. Every piece of the ground truth surface lies within the 3σ range around the estimated mean.

6.6 CCM

The CCM algorithm has been previously described in depth in Chapter 5. Its main benefit over the aforementioned piecewise constant maps is the consideration of slope in the terrain representation.

Roughness (Rocks / small craters) and inclination of the terrain can therefore be evaluated individually. Using the slope corrected variance also allows for sub cell size obstacle detection and does produce a higher precision sample variance aware model of the area.

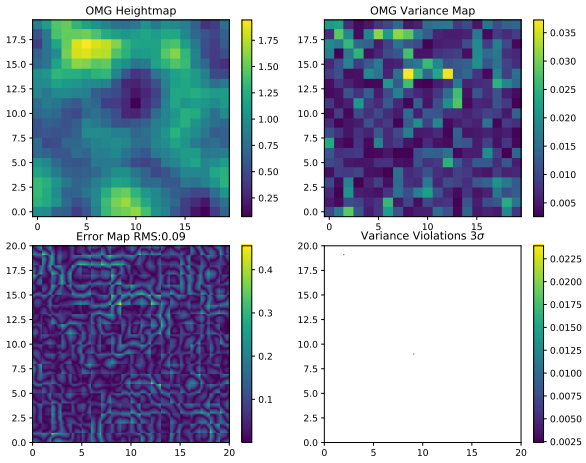


Figure 6.4: OMG Map 1 m Cell Size Evaluation.

6.6.1 *Evaluation*

At first glance, the height map in Figure 6.5 does not differ much from the previous two maps. This is the case because for the pure height map plot only the mean height per cell is used for plotting. For error evaluation, the map is queried on a sub-cell resolution and therefore slope is taken into consideration.

Additionally, the two slope estimates for each cell are plotted for reference. Areas of high inclination around the main crater are clearly visible.

Being a first order approximation of the underlying terrain, the cells can have more than one intersection with the ground truth. Upon closer inspection this is clearly visible in the error map plot as multiple local minima are present within each cell. As expected, the overall error is significantly lower with an RMS error of just 0.028.

Similar to OMG in Section 6.5 this approach also estimates the sample variance correctly. It is worth mentioning, that the total amount of estimated variance is significantly lower compared to OMG. By estimating the variance with respect to a first order approximation only deviations from the smooth inclined surface are contributing to the perceived variance. The variance value is therefore more expressive regarding true surface roughness.

All of this comes at the cost of increased computational and memory usage with 9 float values per cell.

After decreasing the resolution of CCM by a factor of two, the RMS error is in the same magnitude as for the other piecewise constant maps

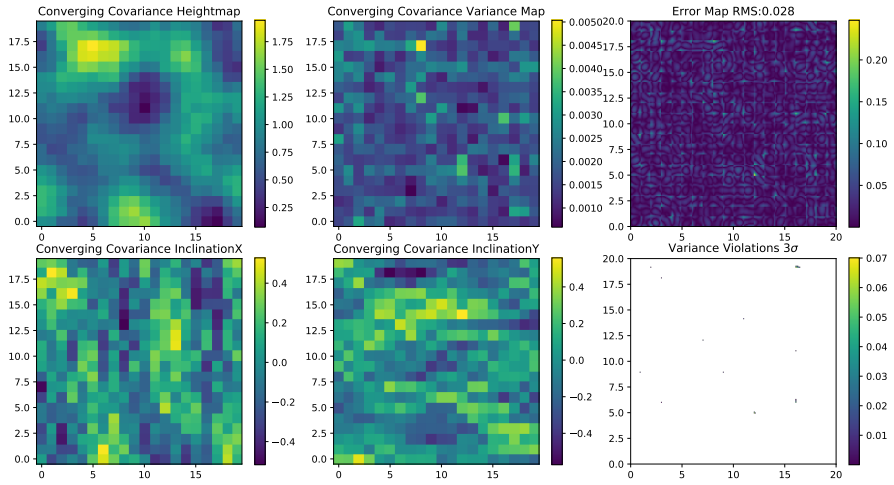


Figure 6.5: CCM Map 1 m Cell Size Evaluation.

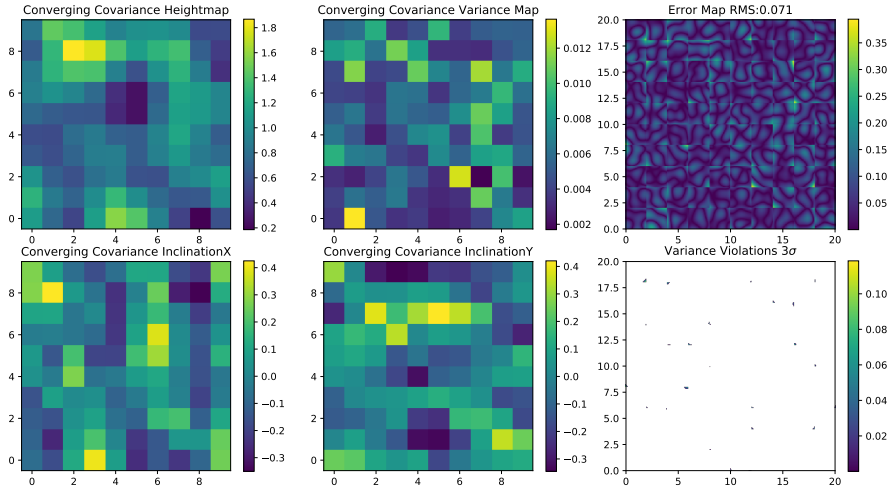


Figure 6.6: CCM Map 2 m Cell Size Evaluation.

maps. This is visualized in Figure 6.6. Therefore, at equal memory usage, CCM still outputs a higher precision DEM.

6.7 HAZARD DETECTION

6.7.1 *Piecewise Constant*

Hazard detection for conventional piecewise constant maps is performed by a simple sliding window roughness test. The DEM is evaluated by moving a circular or rectangular filter kernel sequentially over the map. For each cell, the surrounding cells are evaluated for the maximal and minimal height value. By applying a threshold to the maximal height difference / *roughness* a binary hazard / no hazard classification can be made. A circular roughness filter with a size of 5×5 cells is shown in Figure 6.7.

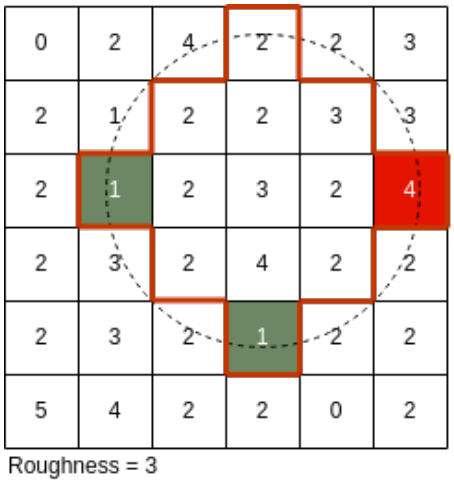


Figure 6.7: Roughness Filter piecewise Constant Map

The map border area, where part of the roughness filter is outside the map margins, is not evaluated at all. This ensures, that enough of an obstacle is visible to set off the roughness test. Later evaluations show a *map border* of unknown labeled area resulting from this constraint.

By introducing sequential buffering, the roughness test can be performed in an optimized manner. Since the map is evaluated row by row with single column increments, a big overlapping area is existent from one cell to the next. With the addition of storing the pixel coordinates of the min and max values, only the non overlapping pixels of the next roughness test need to be evaluated and compared to the old values.

6.7.2 *CCM*

The hazard detection of CCM is presented in 5.9.

6.8 CONCLUSION

Three distinct mapping updates have been presented in detail and were evaluated on simulated data. Kalman and OMG mapping share a common height and obstacle estimation, while the third algorithm is using a different technique. The algorithms were compared based on their ground reconstruction precision.

All three presented mapping updates have distinct benefits and drawbacks. While the presented piecewise constant maps show a disadvantage in precision, they can be updated in significantly shorter time compared to the more complicated CCM. Potential sub-cell obstacle identification based on variance estimation requires either the use of OMG or CCM. The Kalman map with the introduced addition of forgetting forms a solid baseline if only a height map is needed. OMG extends

this height map with information about the sample distribution in the vertical axis. Both maps have equal terrain reconstruction precision.

By estimating a first order fit of the terrain in each cell, CCM shows significantly higher precision at the same cell resolution. The terrain reconstruction precision of CCM is approximately equal to a piecewise constant map of doubled resolution.

Correct terrain variance estimation is shown by both, CCM and OMG.

A final algorithm selection requires the consideration of system parameters additionally to a high-fidelity 3D simulation and real world experiments. The distinguishing features of all three potential mapping techniques were outlined.

7 EVALUATION

This chapter presents and discusses the map evaluation which ultimately lead to the choice of algorithm for the flight software. Based on the requirements and parameters, evaluation tests were conducted in simulation and reality. System parameters such as camera models, frame rate, resolution, baseline, tilt angle, height from ground etc. are chosen to closely mimic the predicted setup on the moon. Evaluations in simulation are conducted to systematically test the performance of the mapping techniques. In particular the smallest resolvable obstacle for each given map level and resolution is experimentally confirmed.

Various edge-case analysis assisted the iterative process of refining the two mapping techniques. Description of all performed edge-case tests would greatly exceed a reasonable length of this chapter and is therefore not described in detail.

Real world tests are performed to validate the soundness of previous simulation based evaluations. Both, obstacle detection performance and reconstruction precision is metered in a real world testing campaign. Additionally, it was checked whether unintended effects or unmodeled factors are present on the real hardware.

7.1 CADRE MISSION PARAMETERS

The expressiveness of simulated results is heavily dependent on accurate system modeling and parameter matching. Table 7.1 lists the CADRE rover parameters and requirements which are relevant for evaluation and parameter selection of the map. These values are representing design goals and parameters from the Mercury7 [1] rovers. The Mercury7 rovers are a set of RnD prototypes for the CADRE mission. Constructed entirely from 3D printed parts and powerful off-the-shelf batteries and motors, Guidance Navigation and Control (GNC) hard - and software can be tested on the real hardware. The presented parameters were also used for simulation to ensure consistency.

1 Each rover inherited his name from one of the 7 astronauts in the Mercury program. Rovers used in this test were named John, Gus, Scott and Ringo.

System Parameters		
Description	Value	descriptor
Camera Model	OV2311	
Camera Resolution	1600 x 1300	
Field of View	90°	fov
Horizontal Focal Length	671	f_h
Pixel Size	3 µm	
Cam Baseline	5 cm	B
Cam Height Above Ground	15 cm	
Stereo disparity error	0.25	m
Driving Speed	4 cm s^{-1}	s
Frame Rate	1 Hz	
Map Size	10 m ×10 m	
Smallest Hazard Footprint	3 cm	
Smallest Hazard Height	3 cm	
Greatest Slope	20°	

Table 7.1: CADRE Parameters

7.2 INTRODUCTION AND SETUP

7.2.1 *Goal of the Evaluations*

The goals of these evaluations are twofold:

- Prove the capability of the presented maps to detect obstacles of required size.
- Compare terrain reconstruction precision and obstacle performance to select a suitable setup for the flight software.

Due to the severity associated with false obstacle classification, evaluation of the obstacle segmentation is focused.

For both, obstacle detection and terrain reconstruction, the created map is compared to a previously generated ground truth map. For the obstacle side, a simple binary traversability map is used as ground truth, while the terrain reconstruction evaluation requires a high resolution DEM. To proof the correctness and usability of the map, the evaluation has to be carried out at all time steps for all layers individually, to ensure the absence of false classifications during operation.

7.2.2 *General Setup*

Since the rover only observes the surrounding world from its position, the right map patch of the ground truth traversability map and DEM has to be extracted for each time step. Map patch selection is performed based on the ground truth pose of the rover. When working with (artificial) pose drift, this ensures that the ground truth reflects correctly the *true* hazards and terrain as observed by the depth sensor. Drift resilience is added later as this is part of the global map merging.

To perform the analysis in the same way for simulation and in the real world, the ground truth reference generation is decoupled from the actual evaluation.

To accommodate for different testing scenarios and sources of data, a common ground truth format needs to be defined. Input data can be e.g. the Gazebo [2] simulator height map, DARTS [3] height map or a Laser scan from the real world.

Traversability maps are either generated from a list of obstacles used in simulation or by post-processing / hand labeling the real world point cloud. The presented setup provides a data conversion for each of the different ground truth sources in height and traversability.

They get translated to the JPL Digital Elevation Model (JDEM) format. JDEM is a simple human-readable single value per pixel map with a header for map metadata. The metadata includes resolution, center position and alignment information. This way, the evaluation is not constrained to maps which include the origin, but can accommodate for example unaligned laser scan data.

Two JDEM files (one for terrain, one for traversability) are used live during mapping to publish ground truth data which matches the mapping output. Therefore, for evaluation in post-processing, maps can directly be compared without the need for further transformations or synchronization. This eliminates issues which arise when evaluating for pose drift. Additionally, the map can be visualized and compared to ground truth live during operation for simplified debugging and evaluation on-the-fly.

Height output is evaluated by calculating the RMS error for each time step. RMS error is only calculated for regions that are covered by the map and not classified as a drive hazard. The comparison is carried out on ground truth resolution to capture effects induced by different terrain representations.

2 A tightly into ROS integrated robotics simulator https://gazebosim.org
3 JPL internal simulation tool

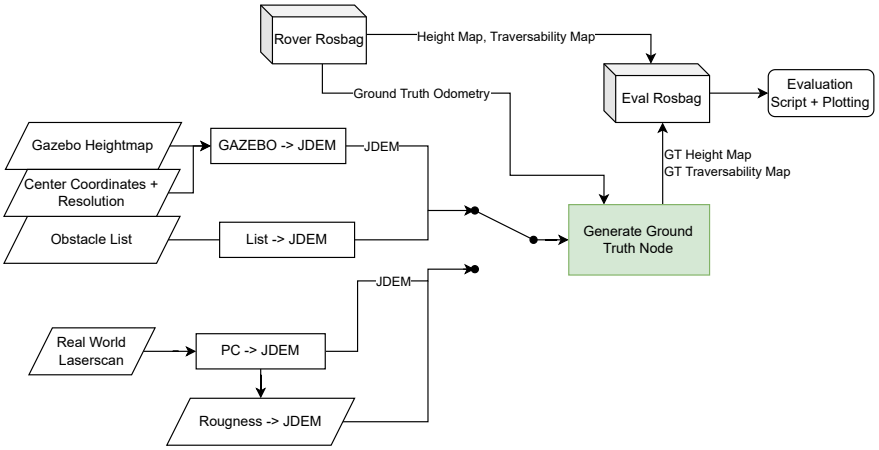


Figure 7.1: Overview of the Evaluation Setup.

The complete described setup is shown in Figure 7.1.

Hazard detection is also evaluated on a pixel by pixel basis. This metric represents how much **area** is correctly classified / false positive / false negative. *Traversability* is used as the nomenclature for false positives / false negatives. A false positive is the worst case, where a drive hazard is mistakenly classified as traversable terrain. The area based metric also shows partially wrong classified obstacles. Partially missed obstacles therefore show up in the evaluation.

7.2.3 *Simulation Setup*

GAZEBO simulator was used for sensor simulation and dataset generation.

As a terrain, a 20 m × 20 m synthetic DEM of a lunar surface with rudimentary texture is loaded. The terrain includes some features such as craters and slopes to allow evaluation of terrain reconstruction precision. Without added obstacles, the terrain is smooth and does not pose any drive hazards. Hazards in form of textured half spheres are added to mimic rocks and boulders embedded into the lunar regolith. Position and size of these obstacles are settable in a single configuration file to enable monte-carlo style evaluations on multiple setups and obstacle sizes. Figure 7.2 shows the gazebo world with 120 randomly placed 10 cm obstacles.

Since this setup is not targeted to evaluate the stereo algorithm itself, textures are only applied to provide detail for the stereo reconstruction. A simple rock texture was therefore sufficient.

Figure 7.2: Gazebo World with 10 cm Obstacles.

The same depth from stereo algorithm *JPLV*, as used in the real mission, is deployed in simulation. Hence, a similar quality and density of the reconstructed depth map can be assumed.

7.2.3.1 *Stereo Camera*

For image generation, a virtual pair of stereo cameras is attached to the *camera dolly*, the only movable object in the simulation. Focal length, stereo baseline, orientation with respect to the terrain and height above ground were set to the flight hardware parameters. This ensures a similar point of view and therefore stereo point distribution as expected in the real world.

To get a similar stereo depth variance as in reality, brightness noise must be added to the simulated images. By capturing a short sequence of still imagery with the real world sensor, this noise parameter is measured. It is to be noted, that the Gazebo camera plugin only models Gaussian noise over the entire image. A more precise model would consider non-linear intensity dependent noise with variable temperature. Ultimately, the available noise model was sufficient for the present analysis. Simulated noise was set to provide a pessimistic approximation of the real images. A snapshot of the image noise analysis is shown in Figure 7.3.

Parallel to the simulation, an instance of *JPLV* creates the depth images and point clouds needed for the mapping algorithms. The ground truth pose of the rover is directly acquired from simulation. To evaluate pure obstacle and height performance, this ground truth pose was directly used for mapping. Later evaluations study the effects of a degraded pose on the algorithms.

For evaluation of the measurement noise sensitivity of different mapping approaches, an additional ground truth depth sensor is implemented in the simulation. This depth sensor has the same resolution

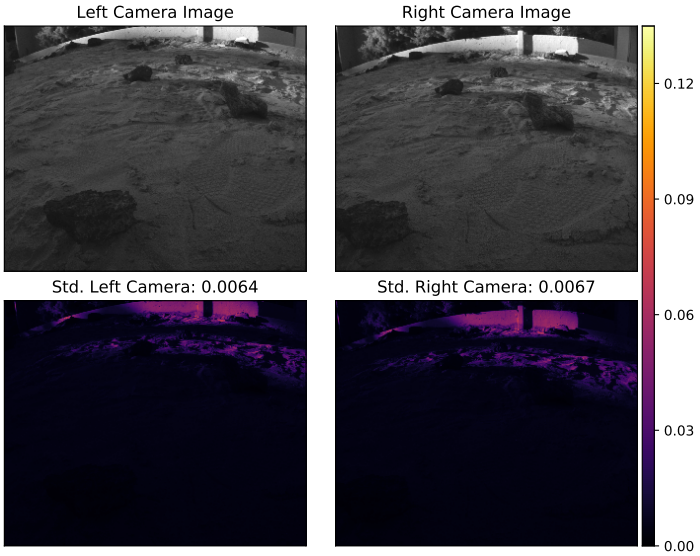


Figure 7.3: Real World Camera Snapshot with Std. Deviation and Heatmap

and Field of View (FoV) as the *depth-from-stereo* image and is mounted at the same location and orientation as the stereo camera. Figure 7.4 shows the two point clouds. The white point cloud is generated from stereo images.

7.2.3.2 *Path Of the Simulated Rover*

The used setup supports two different types of path generation. For in depth analysis of the map performance, a distinct path can be manually recorded and played back. This guarantees the coverage of edge cases in the dataset. The lateral and rotational speed is commanded by the user, while height, tilt and roll angle are dependent on the simulated surface. This way, a realistic view and movement of the cameras are achieved.

Different areas of the surface can either be observed multiple times, just once or even not at all. Those factors can have a significant impact on the performance of a certain map implementation, thus random path generation is an essential part of the testing framework. This is particularly relevant for Monte Carlo simulations and edge case detection.

Path synthesis is performed by selecting random points on the map. They are then connected with cubic interpolation (Splines). Element wise differentiation yields the orientation for each node. Height and orientation are determined similarly to the manual driving by assessment of the terrain beneath. An exemplary traverse path is shown in Figure 7.5.

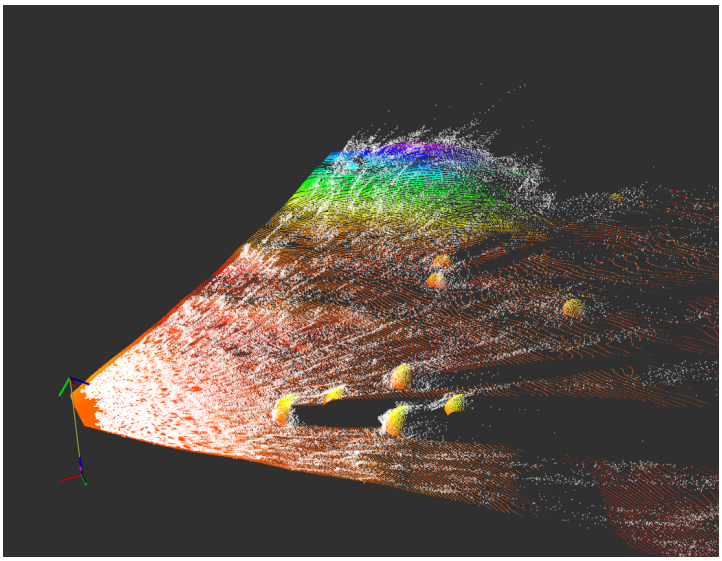


Figure 7.4: Simulated Ground Truth Depth Vs. Depth From Stereo.

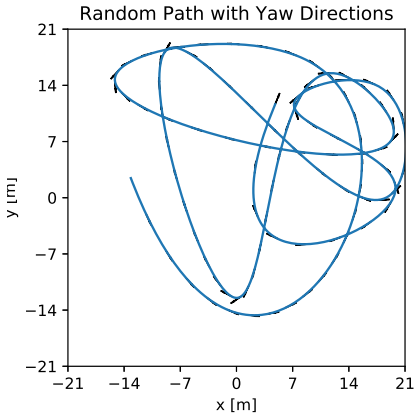


Figure 7.5: Randomly Generated Path.

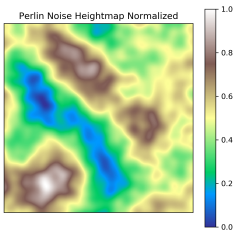


Figure 7.6: Procedurally Generated DEM.

7.2.3.3 *Procedural Height Map*

For testing the algorithm on different terrains, ground truth DEMs must be procedurally generated. This is done by fractal Perlin noise as commonly used in computer game development. Four layers of increasing frequency and decreasing weight are super-imposed to generate a moon-like surface. Variation in frequency, weight and additional features can steer the type of terrain produced. A sample height-map which can be used in Gazebo is depicted in Figure 7.6. A correct selection of weights and frequencies yields a lunar like surface with hills and craters.

Textures do not have to match the terrain and are only used for posing a reliable target for stereo matching. They are either also generated with Perlin noise at a much higher spatial frequency and resolution or simply recycled from moon imagery.

7.2.4 *Real World Setup*

7.2.4.1 *Real World Ground Truth*

Real world ground truth data for the rover position was acquired with a Vicon [4] motion capture setup. Multiple reflective marker on the rover enable external pose determination.

Since for mapping, only the pose of the stereo camera is of interest and the transformation from the robot local Vicon frame to the cameras needs to be calibrated. Vicon2GT [83] is used to calibrate the transformation from the robot local Vicon frame to the Inertial Measurement Unit (IMU) coordinate frame. Kalibr [84] was used further to calibrate the IMU to stereo camera transformation. For evaluation purposes, both transforms were published simultaneous to the robot operation.

The ground truth for obstacles and terrain is acquired from high resolution laser scans. These scans have to be transformed in the same coordinate frame as the Vicon world navigation ground truth. A custom

4 https://www.vicon.com/

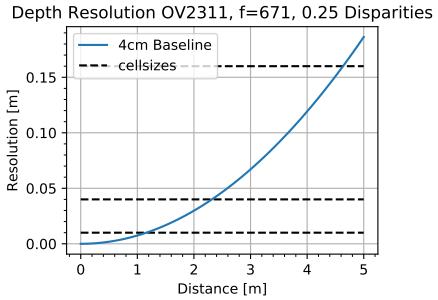


Figure 7.7: Depth Resolution for Camera Setup

alignment process for Vicon position data and high resolution laser scans was developed and successfully applied for this evaluation. Appendix a shows the details of the entire process and acts as a manual for future evaluations.

7.3 PARAMETER SELECTION

Both, CCM and OMG have tunable parameters which need to be set. Parameter selection is carried out entirely on analytic considerations to avoid the influence of assumptions which only hold true on earth. Both maps need parameters for the map size and resolution stack up as well as for the traversability segmentation.

7.3.1 *Piecewise Constant (OMG)*

7.3.1.1 *Layer Sizes*

The most fundamental parameter for any type of multi size multi resolution map is layer count, size and resolution. It is favorable to extend high resolution layers as far as the measurement precision makes it feasible to avoid wasting data or memory / runtime. To ensure, that most of the measurements associated with a cell are actually originating from that cell, a resolution cutoff distance was chosen based on the expected stereo depth precision. As soon, as the expected standard deviation in depth supersedes the resolution, this layer is not continued.

The expected standard deviation and thus maximal resolution of the map for the proposed camera setup on level1 stereo is depicted in Figure 7.7.

As previously shown by [81], obstacles which have a smaller footprint then three times the map resolution can be missed by piece wise constant maps. For the sake of parameter selection, we assume similar height and lateral extension of potential obstacles. The CADRE mission

requires a minimal detectable obstacle size of 3 cm. Consequently, the highest resolution layer must have a resolution of at least 1 cm.

Given the considerations from Figure 7.7, a map size of 128x128 cells was selected for that layer. To avoid redundancy and an unnecessary amount of layers, a 1:16 ratio is used between the layers. Each covered low resolution cell therefore has the same footprint as 4x4=16 cells on the next layer. The selected map sizes for the piecewise constant maps are shown in Table 7.2. Layer 3 satisfies the size requirement of the map and thus marks the lowest resolution layer. Exact map sizes were chosen to be powers of two for a more efficient memory handling.

Piecewise Constant			
Layer	Resolution	Layer Size Cells	Layer Size
0	1 cm	128	128 cm
1	4 cm	64	265 cm
2	16 cm	32	512 cm
3	64 cm	16	1024 cm

Table 7.2: Piecewise Constant Layer Sizes

7.3.1.2 *Obstacle Detection*

To safely detect 5 cm high obstacles, the roughness threshold is selected to be 2.5 cm for the highest resolution layer. Small roughness filter kernels have the tendency to falsely classify non-traversable terrain as hazard free due to possible local flatness. Therefore, the roughness kernel should have at least the size of a wheels' footprint. We chose the highest resolution roughness kernel to have a size of 7×7 cells. This has proven to be a good trade off between added false negatives around the obstacle and robustness in detection.

The roughness filter not only picks up local discontinuities, but is also sensitive to inclination. A 7 cm diameter roughness kernel with a maximal tolerable roughness of 2.5 cm thus implies a slope constraint of 19.7° for the highest resolution. This slope constraint is further used to select the thresholds of the remaining levels. A roughness threshold resembling the same maximal slope is the most aggressive filter tolerable without running into slope induced false detections. Selection of the filter size and roughness constraint thus determines the height of the smallest detectable obstacle for each layer. Table 7.3 shows the selected sizes and roughness thresholds. The minimal size of detectable obstacles is therefore three times the cell size in lateral direction and the roughness threshold value in height.

Roughness Filter			
Layer	Resolution	Filter Size	Roughness Threshold
0	1 cm	7	2.5 cm
1	4 cm	5	7.3 cm
2	16 cm	5	30 cm
3	64 cm	3	70 cm

Table 7.3: Piecewise Constant Roughness Filter Parameters

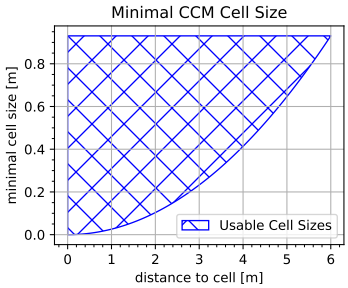


Figure 7.8: Minimal Cell Size CCM over Distance

7.3.2 *CCM*

The parameter selection for hazard segmentation with CCM follows a similar scheme. Previous evaluations have shown, that by utilizing the estimated sample covariance, CCM is capable of detecting sub-cell sized obstacles. To repeatably detect a 3 cm obstacle a minimal cell size of 4 cm was chosen. For evaluation purposes, a level step factor of two instead of four was chosen for CCM. The selected cell sizes therefore are `[4, 8, 16, 32, 64] cm`. Following the same law, detectable obstacle sizes should be `[3, 6, 12, 24, 48] cm`.

Since this approach has tighter constraints regarding acceptable measurement variances, level sizes need to be selected with caution. Measurements must be precise enough to allow for slope reconstruction while estimating the sub cell roughness. This requires a measurement standard deviation much smaller than the cell size. As previously described, for CCM, the measurement variance must be below the expected uniform distribution variance of the cell size for a successful slope reconstruction. Figure 7.8 shows distance / cell size combinations which do not violate this constraint. The expected measurement variance per distance is modeled based on the presented sensor model and camera parameters used for the CADRE mission. Selected map sizes and resolutions are shown in Table 7.4

CCM			
Layer	Resolution	Layer Size Cells	Layer Size
0	4 cm	42	168 cm
1	8 cm	32	256 cm
2	16 cm	28	448 cm
3	32 cm	18	576 cm
3	64 cm	16	1024 cm

Table 7.4: CCM Layer Sizes

With known map level resolutions and dimensions the expected surface normal variance for any given obstacle size is determined. This value is used for the roughness thresholding in the traversability analysis. To estimate the expected variance based on a certain obstacle size and coverage, the two-dimensional sample covariance was simulated. A simple step function with the height of the minimal detectable obstacle represents the obstacle in this simulation. Differently shaped obstacles thus have a degrading impact on detection performance. Since in reality the measurement variance gets added to the sample variance, those effects counteract each other. The presented assumption holds true in the real setup.

Depending on how much of a cell is covered by the obstacle, a different amount of variance is expected. Figure 7.9 exemplarily shows the situation of an obstacle of minimal size covering half of the cell. In Figure 7.10 the development of surface normal variance with respect to cell coverage is plotted. By selecting the threshold variance to be the expected variance at 50% coverage, obstacle coverages from 11% to 91% are detectable. Since perceived variance increases with obstacle size and *shape roughness* of the obstacle, values determined with this technique have shown to be a reliable threshold. This evaluation was done for all map levels.

The inclination threshold is decoupled from the variance and can be set to an equal value for all levels. This value is solely based on rover performance.

Table 7.5 shows the parameters used for the upcoming evaluation.

7.4 EVAL ON SIM

To benchmark the obstacle detection performance of OMG and CCM a Monte Carlo simulation with varying obstacle sizes has been performed. This way a statement about the obstacle detection performance

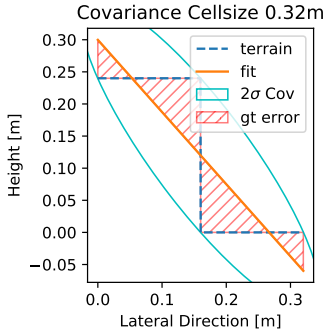


Figure 7.9: 50 % Obstacle Coverage

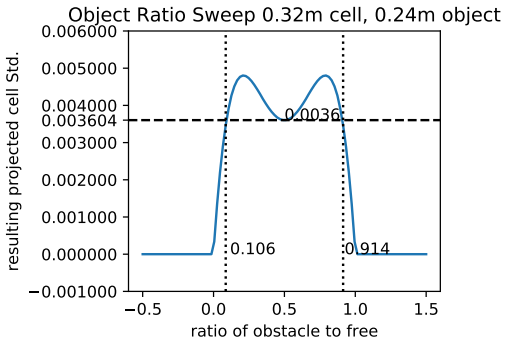


Figure 7.10: Std. Profile Obstacle

Roughness Filter			
Layer	Resolution	Level Size	Std. Deviation Threshold
0	4 cm	43	0.000056
1	8 cm	33	0.00023
2	16 cm	29	0.0009
3	32 cm	19	0.0036
4	64 cm	17	0.014

Table 7.5: CCM Parameters for Obstacle Detection

in correlation to mapping technique and selected layer can be made. Each simulation run incorporated a uniform obstacle size. The used obstacle sizes are:

```
[2  3  4  5  6  8  10  12  14  16  18  20  22  24  26  28  30
34  38  42  46  50  54  58  62  66  70 ]cm
```

Obstacle sizes are the radius of the half sphere as discussed in 7.2.3.

7.4.1 *Piece Wise Constant*

Since the estimated mean per cell and therefore hazard detection performance are equal for OMG and Kalman update algorithms, only one of them needs to be evaluated in terms of hazard detection performance. OMG has been selected without the loss of generality. The OMG obstacle detection performance is plotted per layer in Figure 7.11.

Overall, the following evaluation does not show any unexpected behavior. Small obstacles are only visible in the high resolution layers, while larger obstacles get visible subsequently.

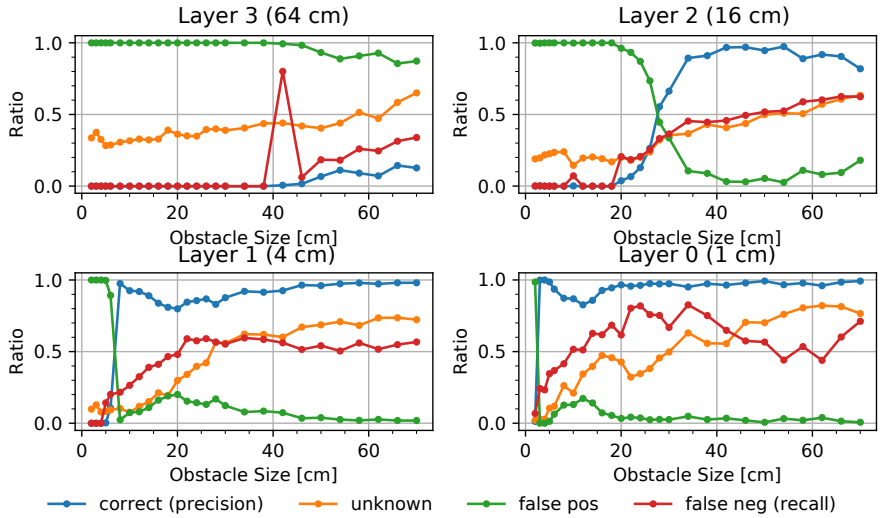


Figure 7.11: OMG Performance for Individual Layers and Obstacle Sizes.

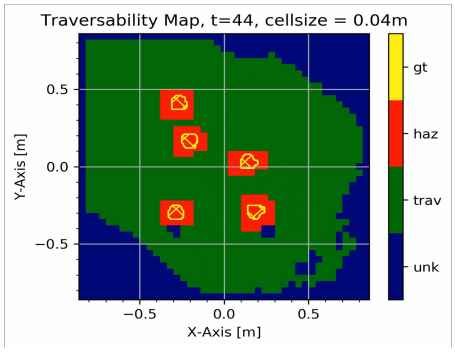


Figure 7.12: Nominal Performance of the OMG Map When Operated Within Parameter Bounds.

The evaluation focuses on edge cases which diverge from the nominal behavior. All other time steps, obstacle sizes and layer combinations perform as expected and yield good results. An example of the OMG map performing nominal is shown in Figure 7.12. All covered ground truth (gt) obstacles are correctly classified as hazardous terrain.

Concluding, one major flaw of piece wise constant maps in the proposed environment is identified. Low resolution layers as necessary for coverage are incapable of detecting obstacles in the applied rover configuration as later presented in Section 7.4.1.5. Therefore, a heterogeneous map setup is proposed for the final product.

7.4.1.1 *2-5cm Obstacles*

By design, 2 cm obstacles are not visible in any layer as shown by the frame by frame analysis in Figure 7.13. For every given time step, correct classifications remain zero while false positives are present depending on weather the map covers obstacles or not. Evaluation is performed

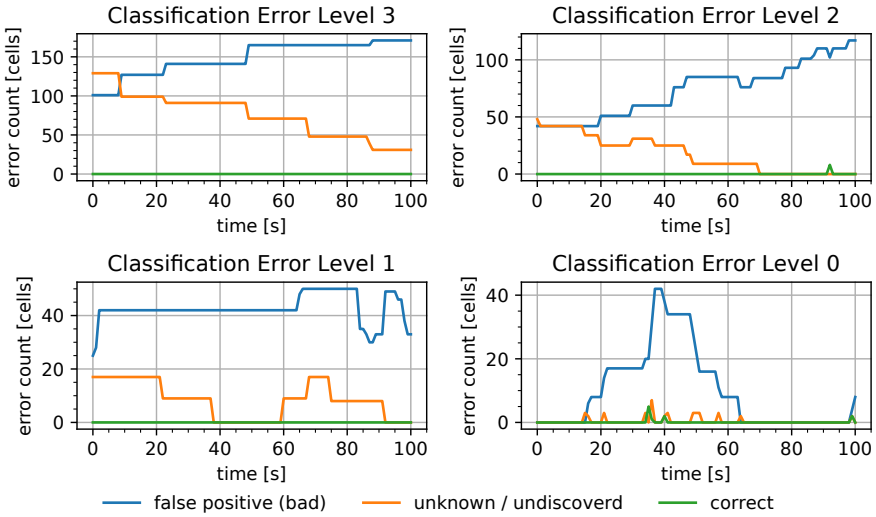


Figure 7.13: 2 cm Obstacle Frame By Frame Analysis OMG.

for each layer individually. No correct classified obstacles are present at any time in any layer.

The situation changes with an increased obstacle size of 5 cm. As expected with the current parameter selection, layer 0 shows no *significant* false positives anymore as visible in Figure 7.14. All other layers do not detect these obstacles and subsequently show a similar result as before. The small false positive rate at 42 s can be explained and further neglected after consideration of the map plot evaluation at this time step in Figure 7.15. All false positives are only occurring at the very top of the (spherical) obstacles, where roughness is low, or the evaluation is done in the shadowed region of the obstacle. False detections on top of obstacles are uncritical since they are not reachable by the rovers' planner.

The false classification in the shadowed region is a product of increased stereo error at the sharp border of obstacles. Due to the very low measurement count in these cells, they instantly converge to the correct value upon *real* observation. As soon, as the rover would be able to traverse this region, it is correctly classified. Both false positive phenomena are thus irrelevant for a safe rover operation.

Additionally, a reasonably sized safety region surrounds the obstacles. In coherence with the parameters selected, the detectable obstacles start at 3 cm, while a *true* false positive rate of 0 is achieved at 4 cm.

The visible map border originates from the roughness filter border condition described in 6.7.1.

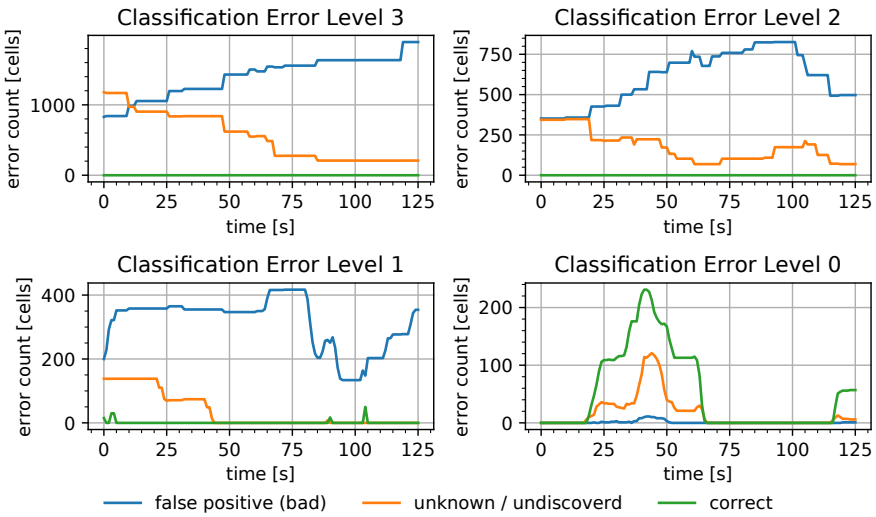


Figure 7.14: 5 cm Obstacle Frame By Frame Analyses OMG.

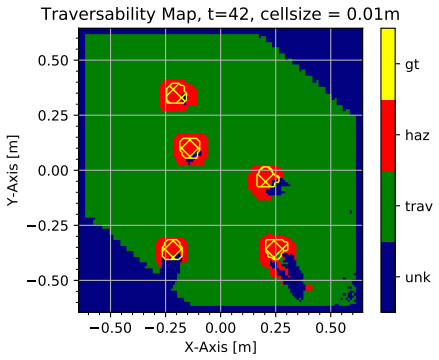


Figure 7.15: OMG Map Plot at T=42 s, Layer 0.

7.4.1.2 *10cm Obstacles*

Advancing further in Figure 7.11, a significant increase in false positives is visible at 10 cm obstacle size. Furthermore, layer 1 is now capable of detecting obstacles. Figure 7.16 shows the frame by frame analysis for the 10 cm obstacle class. Similar to the very small obstacles, layer 0 has some false positives at around 42 s. Upon inspection of each associated map frame, only mid-obstacle cells are falsely classified as exemplarily depicted in Figure 7.17. Hence, this false positive region can safely be neglected.

Layer 1 on the other hand has one true false positive at T=10 s. The situation is shown in Figure 7.18. This case in particular is the first observed edge case. Located at the center, the robot does not have a clear line of sight to the obstacle in question. A different obstacle shadows the wrongly classified obstacle partially. Since only the top of the shadowed obstacle is visible, cells converge only to the top value. Stereo noise spreads this value to neighboring cells, creating a plateau at obstacle height. The roughness filter is incapable of distinguishing this effect from a traversable surface. To better illustrate this situation Figure 7.19 shows a three-dimensional representation of the map and point cloud from stereo. It is visible, that the stereo error at the top of the front obstacle bridges the gap to the obstacle behind. This false positive resolves as soon as the obstacle is in plain sight and therefore not an issue for the safety of the rover. The false positive starting at T=70 s is a similar issue with half of the obstacle hiding behind a hill of the terrain.

The map's ability to re-converge to the correct value is dependent on the forgetting parameter, which should be tuned accordingly. Variance / S-value inflation and convergence limiting steer the adaptability.

7.4.1.3 *20cm Obstacles*

The next interesting obstacle size in Figure 7.11 is 20 cm. Figure 7.20 shows the frame by frame analysis for 20 cm obstacles. Layer 2 slowly starts to pick up individual obstacles due to the perceived roughness increase from nearby slopes.

The constant false positive rate over the entire evaluation on layer 1 results from the known low height variance in the center of an obstacle. Upon closer inspection are all obstacles at all times steps correctly classified as exemplary depicted in Figure 7.21. No time step shows a concerning false positive which would result in the rover driving into hazardous terrain.

Layer 0 shows no false positives at all. At this obstacle size only the border of an obstacle is observed at all. The size of the roughness filter

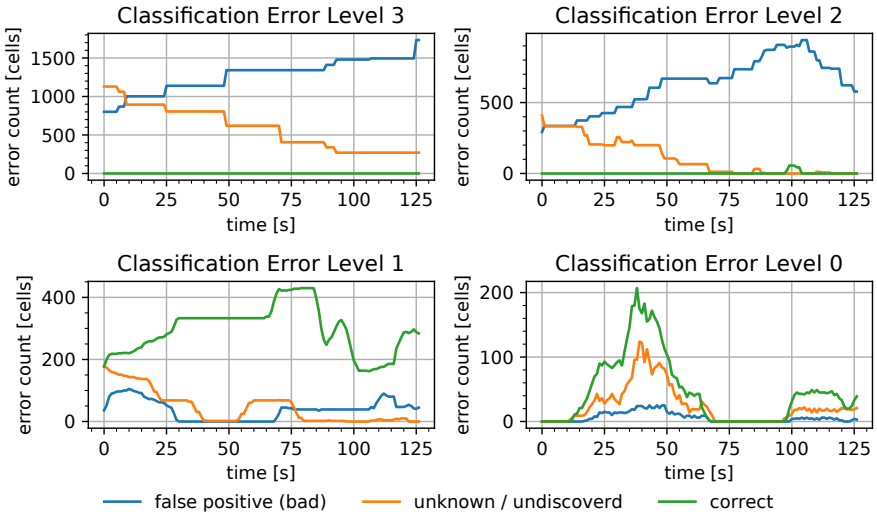


Figure 7.16: 10 cm Obstacle Frame By Frame Analysis OMG.

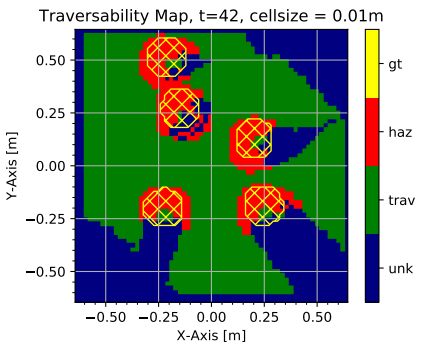


Figure 7.17: OMG Map Plot at T=42 s, Layer 0 with 10 cm Obstacles

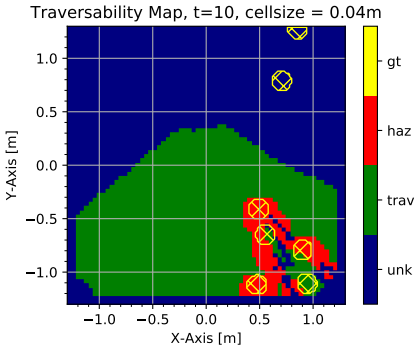


Figure 7.18: OMG Map Plot at T=10 s, Layer 1 with 10 cm Obstacles

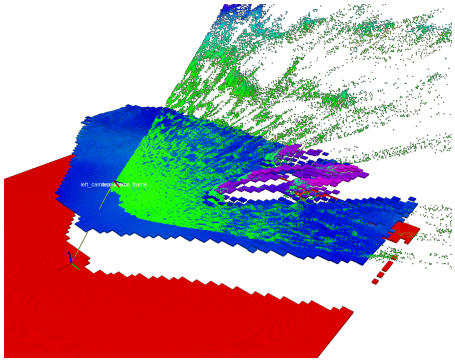


Figure 7.19: Map Snapshot OMG Layer 1 with 10 cm Obstacles at T=10 s

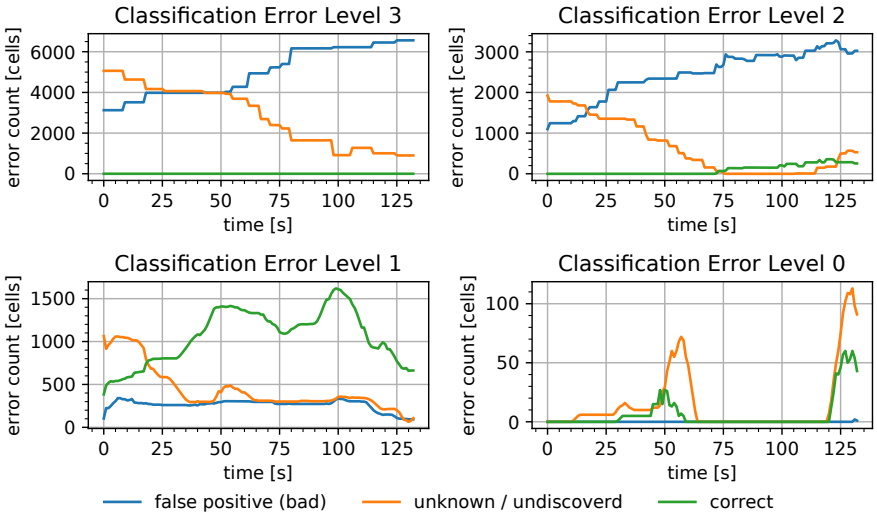


Figure 7.20: 20 cm Obstacle Frame By Frame Analysis OMG.

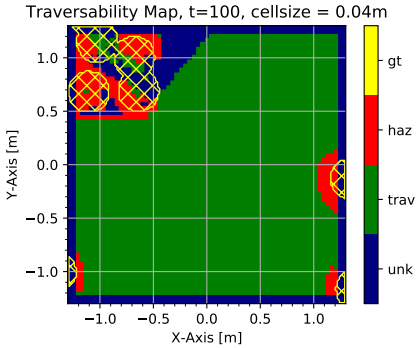


Figure 7.21: OMG Map Plot at T=100s, Layer 1. with 20 cm Obstacles

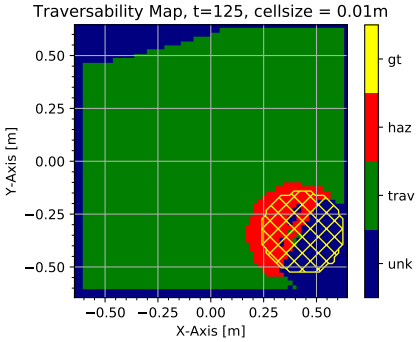


Figure 7.22: OMG Map Plot at T=125s, Layer 0. with 20 cm Obstacles

keeps a safe margin around the border as shown in Figure 7.22. This safety margin in combination with unobserved areas within the obstacle artificially decreases the resulting recall. No regions with over proportional many false negatives were observed.

7.4.1.4 *34cm, 42cm Obstacles*

By parameter selection, layer 2 should be able to resolve 34 cm obstacles. The time analysis in Figure 7.23 shows a significant false positive rate, which peaks at T = 100 s. Closer inspection of this layer and timestamp combination is shown in Figure 7.24. No new issue besides the ones related to other obstacles and obstacle center roughness are visible. It is to be mentioned, that the false negative rate also increased. The area around obstacles is heavily effected by increased stereo variance.

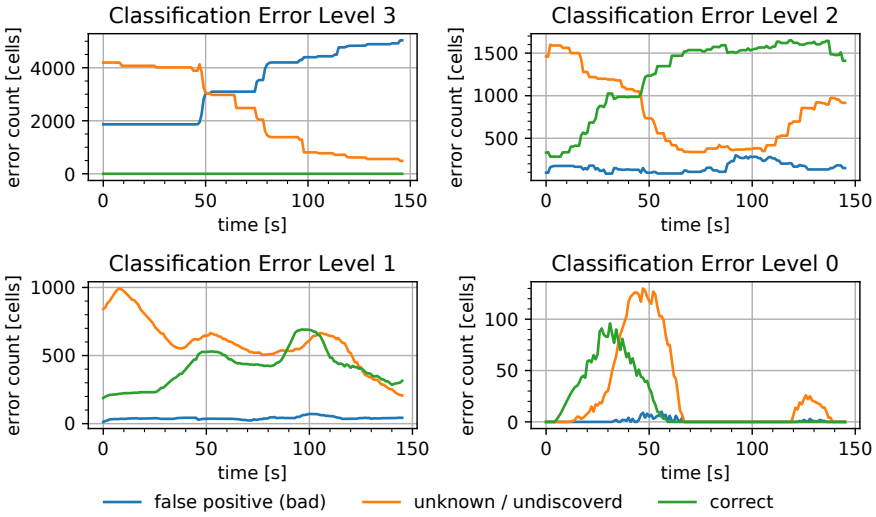


Figure 7.23: 34cm Obstacle Frame By Frame Analysis OMG.

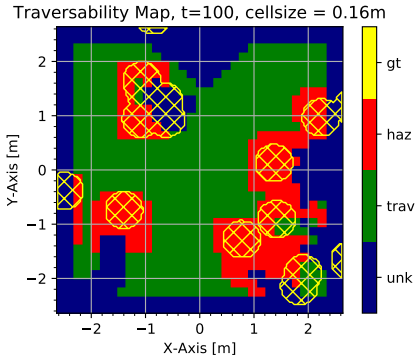


Figure 7.24: OMG Map Plot at T=100s, Layer 2. with 34cm Obstacles

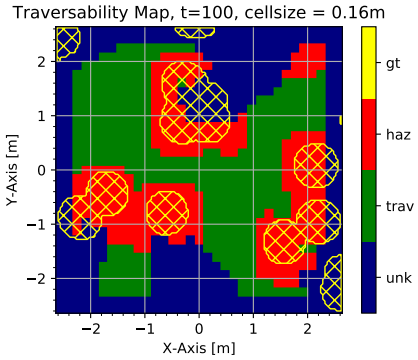


Figure 7.25: OMG Map Plot at T=100s, Layer 2 with 42cm Obstacles

The low measurement per cell count of cells at the border of the line of sight behind obstacles cause significant patches of false negative classification. This artificially increases the perceived size of the obstacle. The rogue points and their affected cells are visible in 3D at the same timestamp as illustrated in Figure 7.26.

With slightly larger obstacle size, those issues become less present as shown at the same time step but an obstacle size of 42 cm in Figure 7.25. Overall, these regions get resolved at the moment of better sensor coverage.

7.4.1.5 *Wall Problem*

The wall problem describes the reason why the lowest resolution layer does not pick up obstacles independent of their size. In particular when encountering big obstacles, the observed surface is mostly vertical. Due

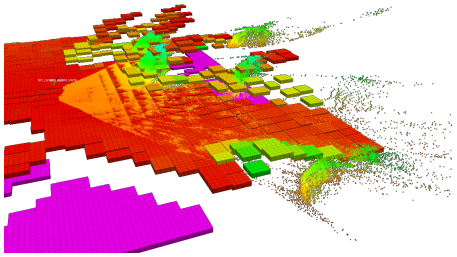


Figure 7.26: Map Snapshot OMG Layer 2 with 34cm Obstacles at T=100s

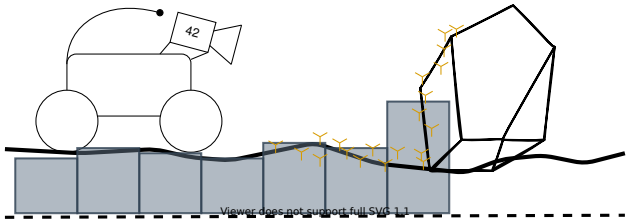


Figure 7.27: Bordering Cells Only Converge to Half of the Obstacle Height.

to the perspective of the rover, only one face of the obstacle is observable while other faces are shadowed. On the lowest resolution, typically all obstacle related measurements are associated to a single row of cells. If the obstacle doesn't align with the cell border also some surface related measurements are associated with those cells.

Since measurements are present at all heights over the vertical surface, the resulting cell height will be at most half of the obstacle height. Figure 7.27 illustrates this issue. To reliably detect an obstacle, obstacles would have to be at least two times the height of the roughness filter.

Visibility constraints of the camera system amplify this issue. At the CADRE rovers, the cameras have a field of view which tilts 10° above the horizon. The cameras itself are mounted 15 cm above ground. As illustrated in Figure 7.28, this results in 102 cm visible height at a maximum range of 5 m. In the best case, this would result in a cell height of 51 cm which is well below the roughness filter of 70 cm.

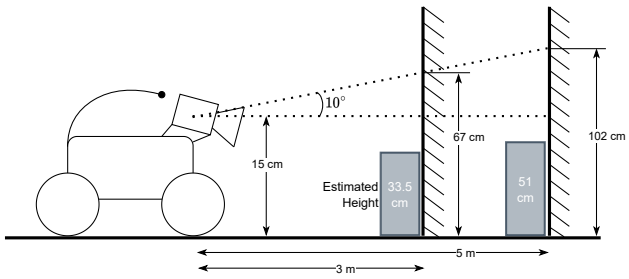


Figure 7.28: Rover Setup does not Allow Obstacle Detection on the Corse layer.

As a reminder, the roughness threshold of 70 cm is implied by the 20°slope constraint and can therefore not be reduced.

Consequently, the most corse layer is only capable of detecting non-shadowing inclines, which allow observation due to the rover being tilted itself. Large rocks are thus only detected in edge cases.

7.4.2 *Conclusion*

Over all, traditional piece wise constant maps such as the Kalman map or the OMG map have proven to be a viable choice for the CADRE use case. The high resolution layers detect the designed obstacles reliable with expected side effects. All observed false classifications were traceable to known effects and do not pose a hazard for the rover. Low resolution cells require big obstacles for correct classification. The proposed lowest resolution layer is incapable of detecting obstacles with the given robot parameters. A pure piecewise constant map is therefore not suitable for CADRE. The benefits lie in the nature of simplicity and reliability.

7.4.3 *CCM*

The CCM layer performance is plotted in Figure 7.29.

Similar to the previous piece wise constant map evaluation, distinct tipping points are visible for each layer.

Again, this evaluation focuses on the challenges and issues evident in the map. The majority of time steps, layer size and obstacle combinations show very good obstacle detection performance. A typical CCM classification is shown in Figure 7.30.

7.4.3.1 *2cm Obstacles*

Unsurprisingly the frame by frame analysis in Figure 7.31 shows a poor obstacle detection performance on all layers. Only level 0 shows a few detections with many false positives. An exemplary snapshot at T =40 s is shown in Figure 7.32. Even though level 3 should not detect any obstacles at this point, some true positives show up at around 70 s. This is caused by a false negative patch accidentally covering the area of an obstacle. The false negative patch in Figure 7.33 is caused by a major increase in variance for the corresponding cells due to stereo artifacts. By looking at the 3D situation in Figure 7.34 the cause becomes vivid: A slight hill covers the better part of the cells while introducing major stereo error induced variance at the rim. Only a few frames later, proper observation of the cells clears the mistake and mitigates the effects from the stereo variance. The map quickly converges to the true value.

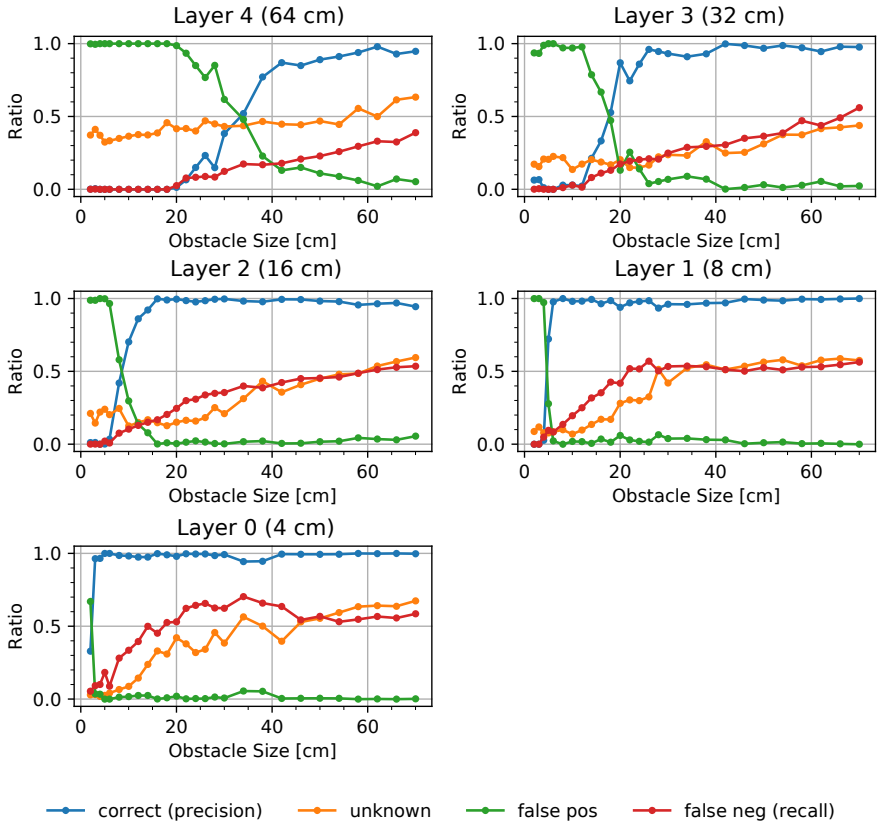


Figure 7.29: CCM Performance for Individual Layers and Obstacle Sizes.

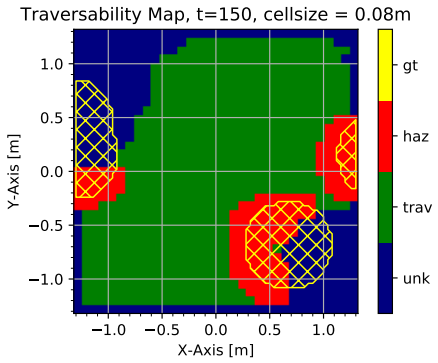


Figure 7.30: CCM Map Plot at T=150 s, Layer 1, 42 cm Obstacles.

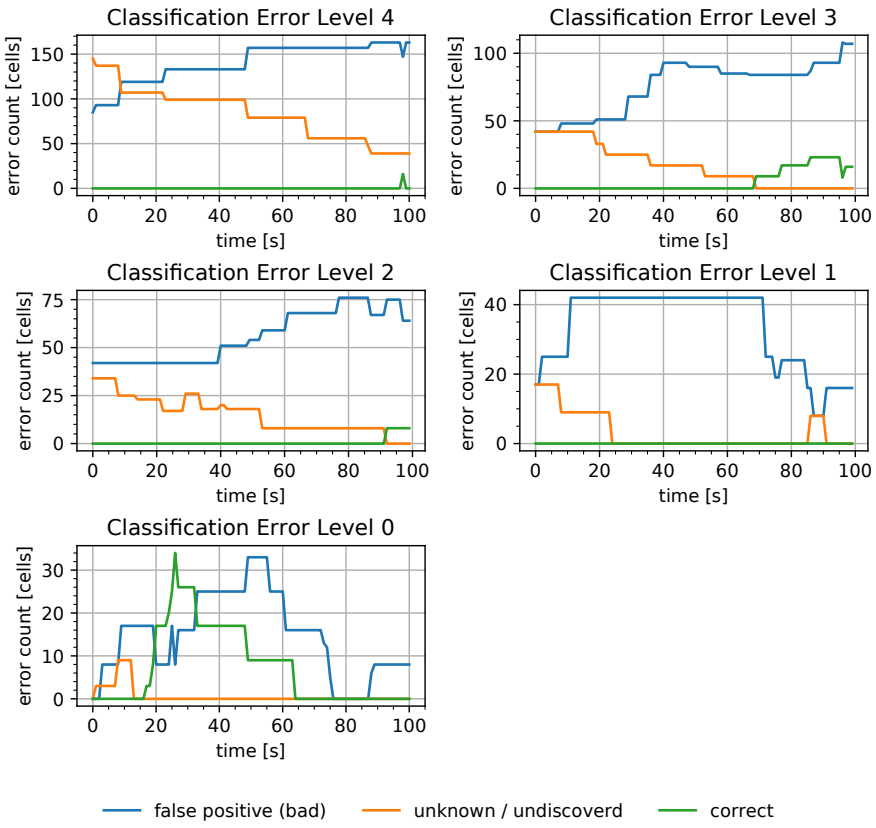


Figure 7.31: 2 cm Obstacle Frame By Frame Analysis CCM.

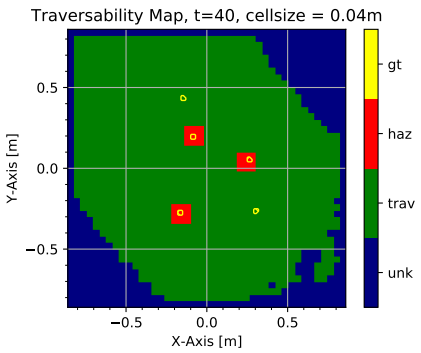


Figure 7.32: CCM Map Plot at T=40 s, Layer 0.

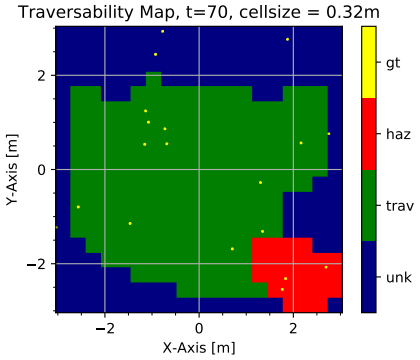


Figure 7.33: CCM Map Plot at T=70 s, Layer 3.

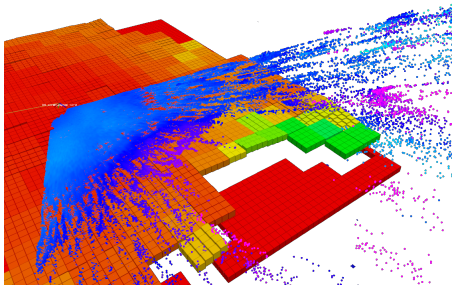


Figure 7.34: Map Snapshot CCM Layer 3 with 2 cm Obstacles at T=70 s

7.4.3.2 *4cm Obstacles*

At 4 cm obstacle size level 0 does not have any false positives or false negatives of significant size. Figure 7.35 displays the individual layer analysis. The one spike of false positives at T=10 s is caused by a hill covering the obstacle. Immediately when the obstacle is properly covered at T=15 s the obstacle is correctly classified.

7.4.3.3 *6cm Obstacles*

At 6 cm obstacle size level 1 is nearly completely settled and only has one erroneous detected obstacle right at the beginning. This obstacle again is located on the distant side of a hill and therefore only barely triples the variance threshold designed for 6 cm. Figure 7.36 shows the time of false positives. Other than that, the layers behave as expected.

Already at the next evaluation step with 8 cm obstacles, no false positives are recorded on level 1.

7.4.3.4 *12cm Obstacles*

12 cm marks the designed detection threshold for layer 2. Visible in Figure 7.37, unsurprisingly layer 1 and 0 show a consistent high performance. The same shadowing which we have already seen at similar sizes in the piecewise constant map also affects CCM. A map plot is shown in Figure 7.38.

Layer 2 on the other hand still has some false positives. Besides the known edge case of shadowed obstacles, one obstacle at T = 69 s does not get detected. The map is visualized in Figure 7.39. In this case, the obstacle is only visible at the far right end of the FoV. After the initial miss classification, the cell does not get observed again to correct the error. Similar cases were resolved at different instances with simply a few more frames on the obstacle.

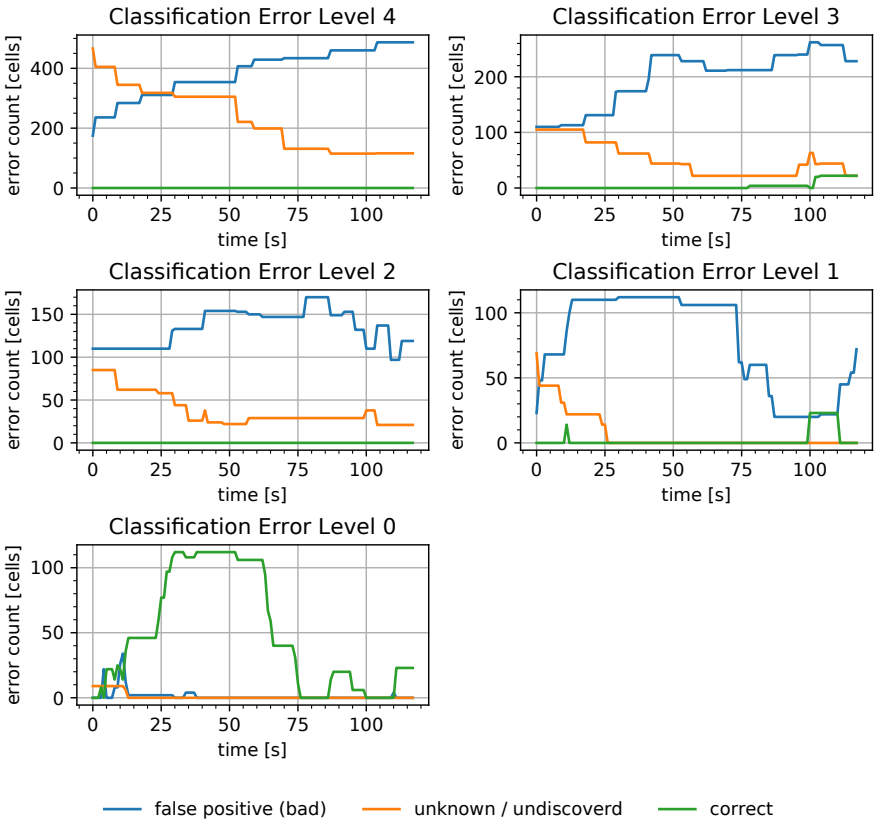


Figure 7.35: 4 cm Obstacle Frame By Frame Analysis CCM.

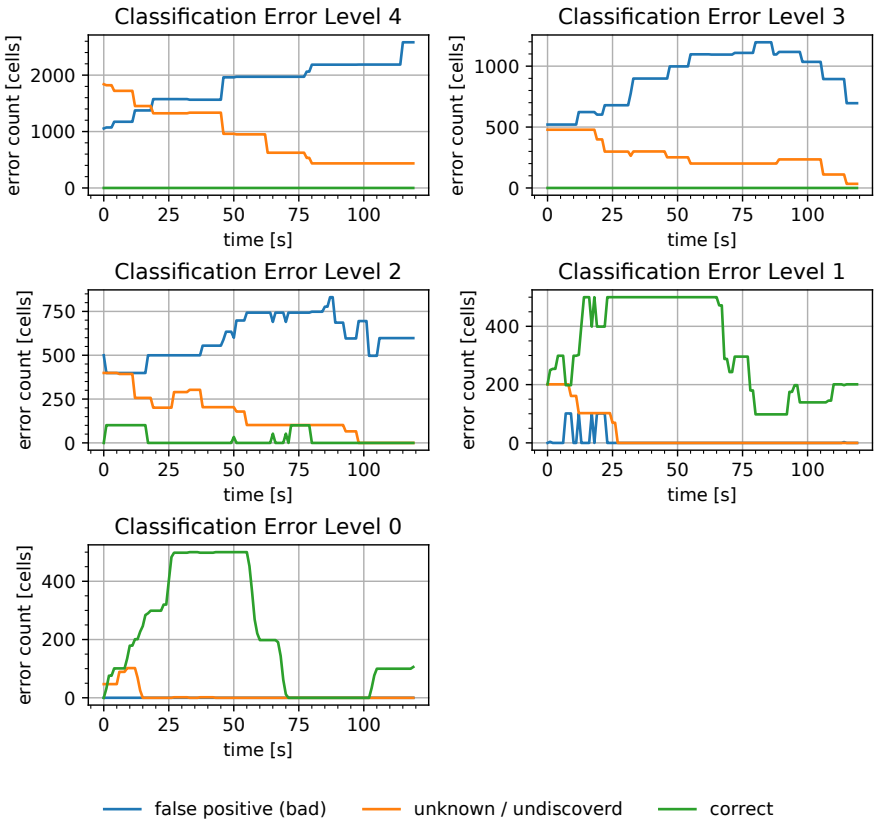


Figure 7.36: 6 cm Obstacle Frame By Frame Analysis CCM.

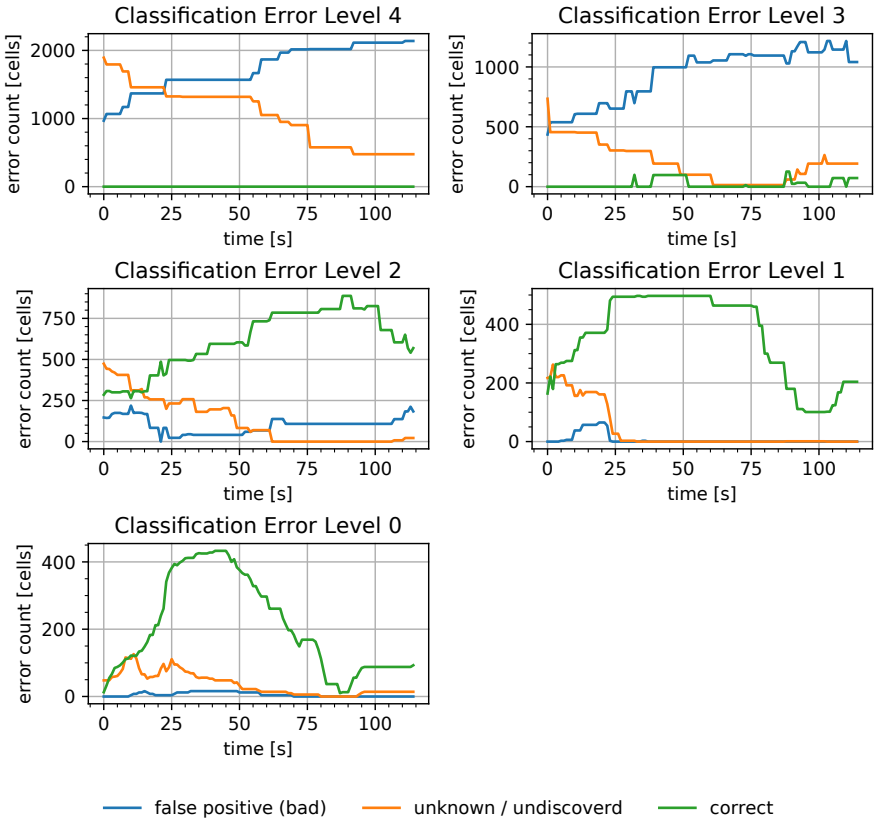


Figure 7.37: 12 cm Obstacle Frame By Frame Analysis CCM.

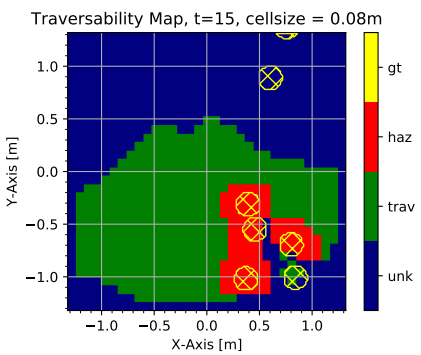


Figure 7.38: CCM Map Plot at T=15 s, Layer 1.

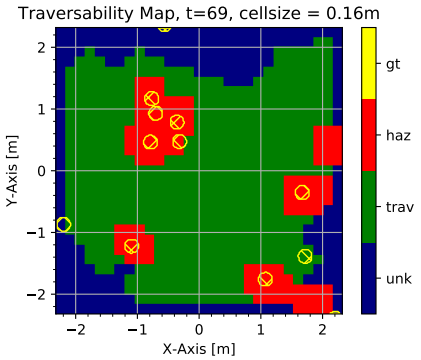


Figure 7.39: CCM Map Plot at T=69 s, Layer 2.

The significant amount of false negatives in front of each obstacle is caused by the same stereo noise effects which also influenced the piece wise constant maps in a negative way. Stray points far from the ground truth surface artificially increase the perceived sample variance of the obstacle-bordering cells. This effect is visible over all map resolutions and obstacle sizes.

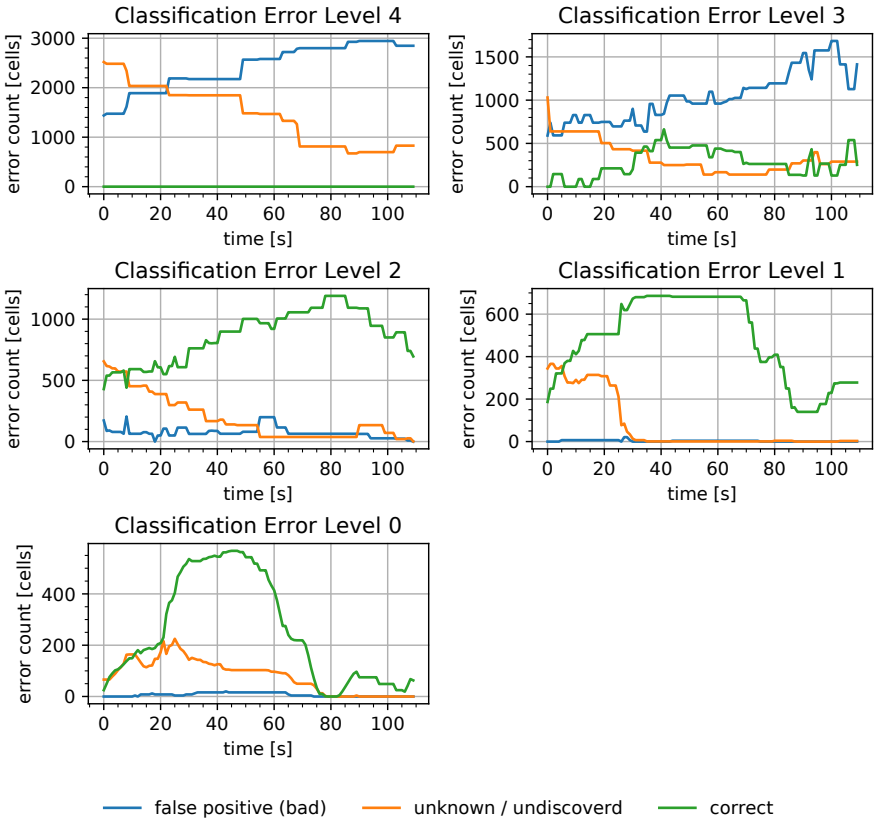


Figure 7.40: 14 cm Obstacle Frame By Frame Analysis CCM.

7.4.3.5 *14cm Obstacles*

Similar to the previous size steps, the issues present at 12 cm vanish as soon as we encounter slightly bigger obstacles. The analysis is shown in Figure 7.40. All remaining false positives on level 2 are caused by shadowing and can safely be ignored.

7.4.3.6 *24cm Obstacles*

24 cm obstacles should be the first obstacle size visible in level 3 by design. Indicated by Figure 7.29 this size still induces false positives on level 3. The higher resolution layers in Figure 7.41 do not show any sign of errors.

The observed false positives are occurring on partially covered obstacles such as the one shown in Figure 7.42. All observed problems are resolved at the next evaluation step 28 cm, as exemplarily shown in Figure 7.43.

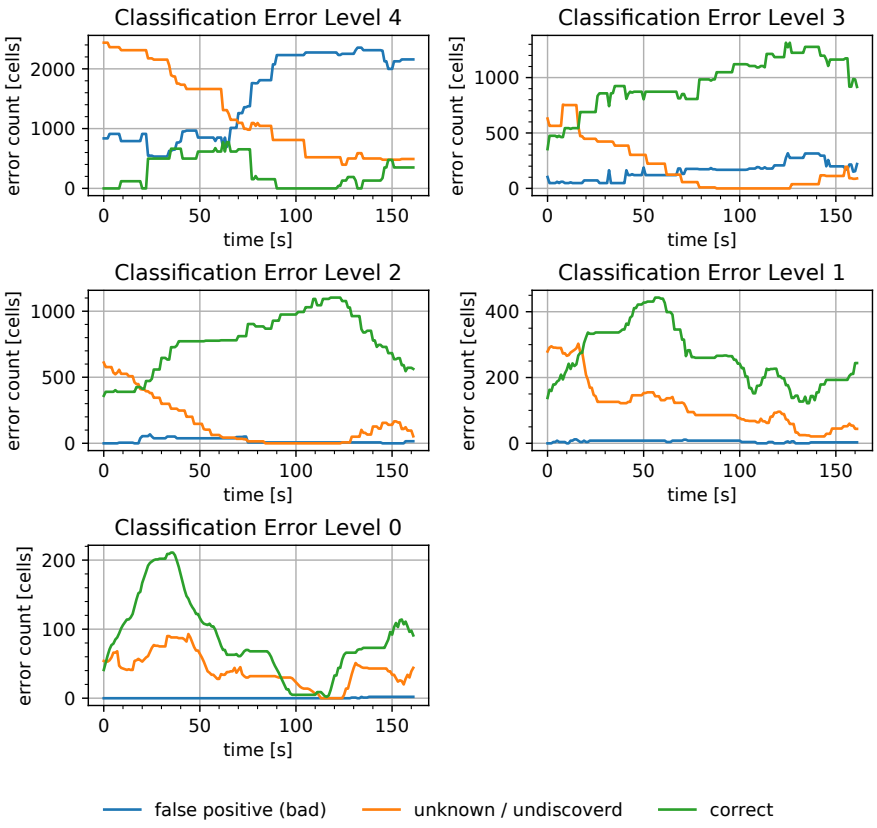


Figure 7.41: 24 cm Obstacle Frame By Frame Analysis CCM.

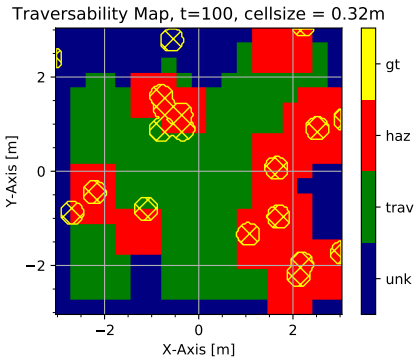


Figure 7.42: CCM Map Plot at T=100 s, Layer 3.

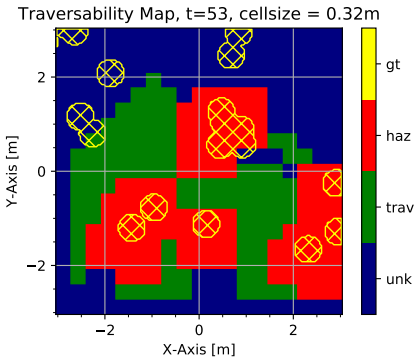


Figure 7.43: CCM Map Plot at T=53 s, Layer 3, 28 cm Obstacles.

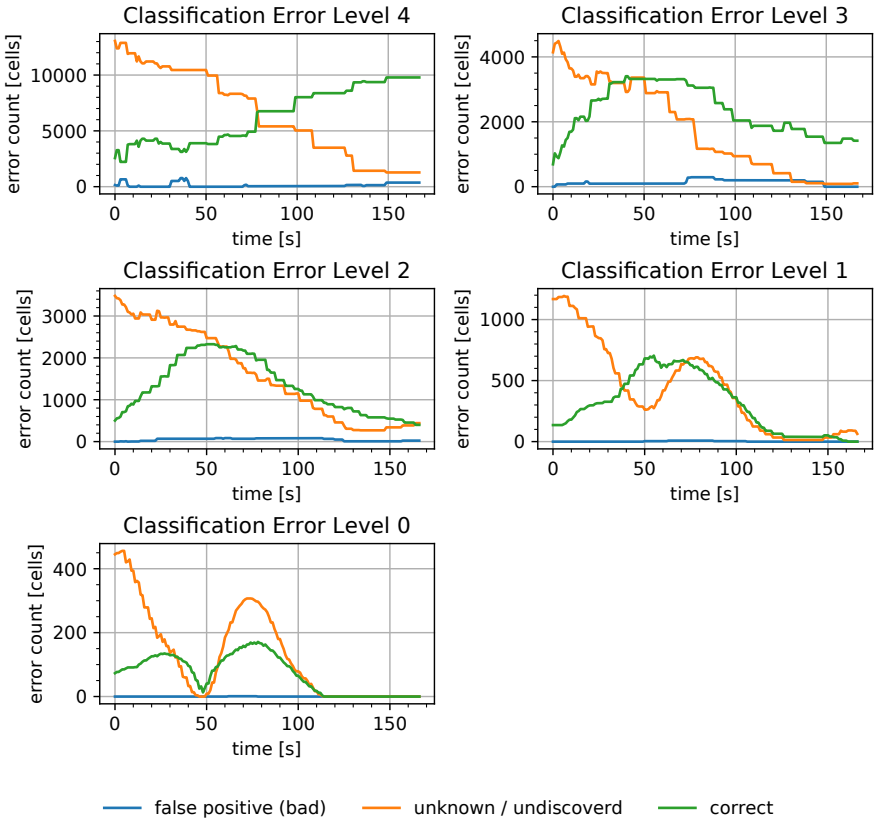


Figure 7.44: 62 cm Obstacle Frame By Frame Analysis CCM.

7.4.3.7 *Bigger Obstacles*

Other than the piece wise constant maps, CCM is capable of resolving even bigger obstacles at the most corse resolution. 62 cm obstacles for example have a false positive rate of zero among all levels as indicated in Figures 7.44. These big obstacles in combination with their high density in the simulation scenario create a canyon like driving scenario. Figure 7.45 shows this situation from the rover's perspective. Stereo noise can close the small passages between obstacles, especially when first observed in a perpendicular direction. This leads to overall worse recall (increase in false negative detections). Figure 7.46 is a plot of the resulting map on level 4. With only a few big obstacles present, even the level 4 performance is similar to the previous layers. Other, then with piece wise constant maps, the lowest resolution layer in CCM can have useful information in case of big obstacles or craters.

The higher resolution layers expectedly perform well no matter what size the obstacle has. As for example level 1 in Figure 7.47 shows the good obstacle detection performance for higher layers. When piece wise constant maps loose wall obstacles at close distance, the CCM map remains stable.

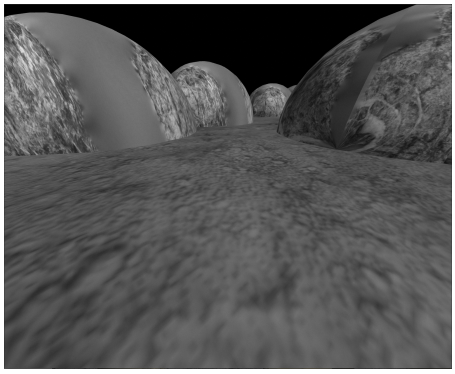


Figure 7.45: Camera Snapshot with 62 cm Obstacles

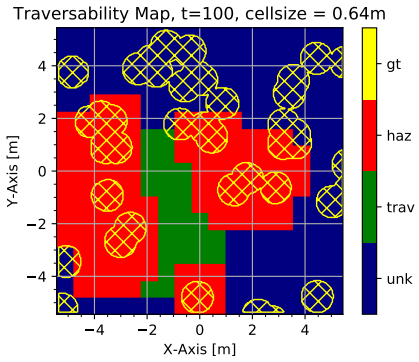


Figure 7.46: CCM Map Plot at T=100 s, Layer 4.

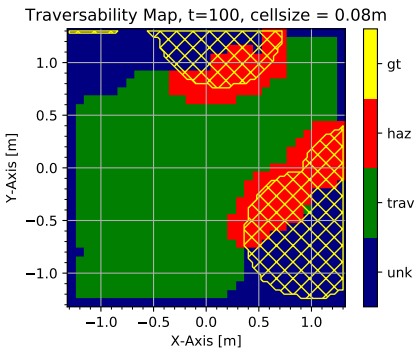


Figure 7.47: CCM Map Plot at T=100 s, Layer 1.

7.4.3.8 *Conclusion*

From the presented data we conclude that the CCM approach together with the present parameter selection and obstacle identification is capable of delivering the needed obstacle precision for CADRE. Based on simulated data, CCM has proven to be a valid candidate. No concerning edge cases or failure modes were uncovered during the evaluation on simulation.

7.4.4 *Height Precision Evaluation*

The maps can not only be evaluated based on their obstacle detection performance but also regarding their absolute terrain reconstruction precision. Similarly to the previously presented rudimentary evaluations on a highly abstracted measurement vector, the individual maps are compared to the ground truth height map in a fine grid. A RMS error can be calculated for each individual map in the time series.

Since particularly rocky surface features can increase the map error when covered (high frequency surface patches) the error is not constant over time.

For evaluation purposes a new ground truth terrain is generated based on Perlin Noise as described in 7.2.3.3. This ensures the presence of both high and low frequency features in the test area. No obstacles or artificial roughness is added to the ground truth. This limits the evaluation to those areas, which are later interesting for map fusion to form a global map.

The same simulated camera rig as used for the previous analysis is reused for consistency. A medium length (80 s) drive path was recorded to cover some evaluation area. It is ensured, that the ground truth area is much larger, then the maximum map size to allow analysis down to the lowest resolution layer.

The following RMS values do not have absolute meaningfulness but should be interpreted relative between the layers and mapping approaches.

7.4.4.1 *Results*

Again, without the loss of generality the OMG update algorithm was used for this analysis. Since Kalman and OMG do not differ in the resulting mean (height), analysis results are identical.

Figure 7.48 shows the precision evaluation for OMG. A distinct increase in precision with map resolution is evident. This can intuitively be

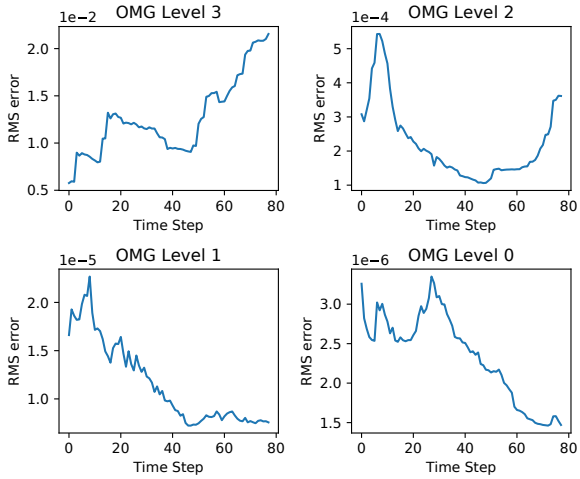


Figure 7.48: OMG Terrain Precision Analysis on Simulated Data.

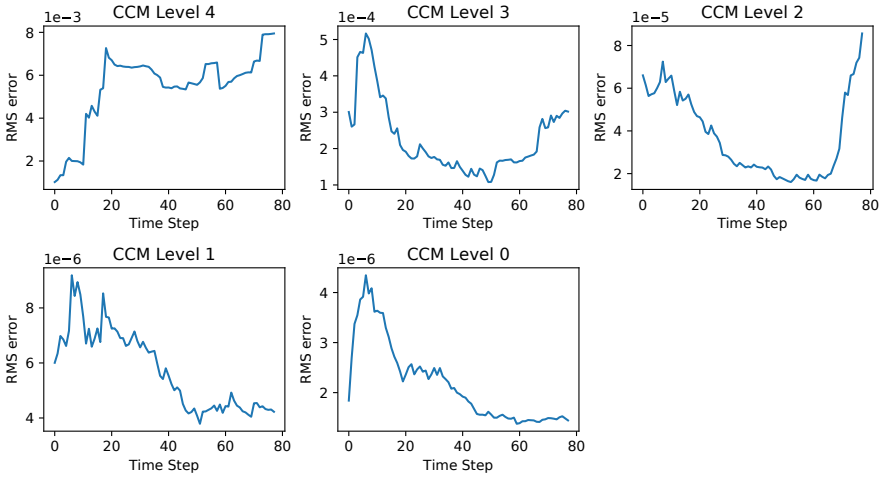


Figure 7.49: CCM Terrain Precision Analysis on Simulated Data.

explained by the higher fidelity of high resolution layers and more precise measurements close to the rover.

Figure 7.49 shows the same evaluation but for the CCM algorithm. Again, higher resolution levels produce higher precision outputs.

Similar to the rudimentary pre analysis in Chapter 6 we can confirm the performance increase of CCM. CCM shows at least the same reconstruction precision as a piecewise constant map of double the resolution.

7.4.5 *Pose Drift Resilience*

On the moon, the real rover will not have the luxury of an external reference pose estimation. Inside out tracking based on VIO together

with a sun sensor and Ultra Wide Band (UWB) range finding is used instead. Even though low pose drift and noise is expected, the remaining error needs to be tolerated by the map.

Conventional piecewise constant maps are comparably insensitive to pose drift and noise. The locally restrained nature of the presented robot-centric map helps to mitigate the effect of pose drift. Relative to the robot, features in the map are known precisely. Only the global representation severely lacks precision when pose drift accumulates.

When the height map gets used for hazard avoidance, it is of great importance for planning, that the *true* obstacle does not leave the marked area even when it is not observed at the moment. A simple calculation based on the added security margin s [5] and map size M reveals the largest compensable pose drift fraction δ_{max}:

$$\delta_{max} = \frac{s}{M} \tag{7.1}$$

Due to the map's ability to forget and converge to new measurements, this estimate is rather conservative. A well tuned maximum integration parameter keeps the map adaptable enough to avoid obstacles in the field of view, even when they appear at already converged cells.

7.4.5.1 *CCM*

Since the perceived sample variance is used for obstacle detection in CCM, a more in-depth sensitivity analysis has to be performed. Added pose drift increases the perceived variance in cells significantly and can cause false obstacle detections. This only results in false negative detections and makes the algorithm more sensitive. No previously detected obstacles will be missed due to pose drift or noise as long as the constraint from the previous section is not violated.

Quantifying the effect of pose drift requires careful selection of the test environment. A simple planar environment does not result in an increased sample variance under added pose drift. The worst case is a terrain which changes inclination while staying below the drivability threshold for slope. A simple sine wave in one direction satisfies this requirement. Changing slopes are offset by pose drift as seen from the rover's perspective. The terrain used for this evaluation is displayed in Figure 7.50.

Pose drift resilience is tested by running the CCM algorithm multiple times on the same dataset with increased pose degradation. The pose

5 In case of a roughness filter for obstacle detection, half the size of the filter is a conservative estimate.

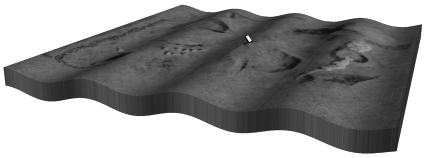


Figure 7.50: DEM Used for Pose Drift Analysis

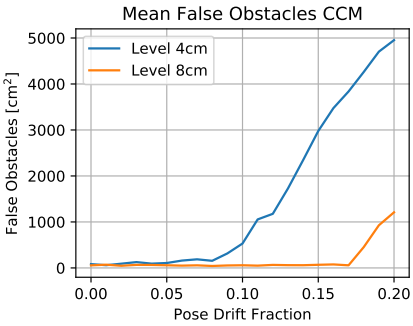


Figure 7.51: CCM Pose Drift Analysis

offset to ground truth is calculated as a function of traversed distance to simulate drifting. 0% to 20% pose drift are tested.

Similar to the piecewise constant maps, the performance is heavily relying on driving speed, frame rate and speed of forgetting / convergence limit. Since this evaluation was mainly conducted to proof, that there is a parameter set which will work in reality, only one set of parameters was tested. Driving speed and frame rate are set to the real world values while the integration limit is set equal to the previous tests.

The evaluation in Figure 7.51 shows, that up to a drift fraction of 10% all layers are intact. Until 10%, no additional false negatives are present on any layer. This suffices, given the expected precision of the localization. Final values for the integration limiter need to be tuned on the real hardware.

7.4.6 *Comparision*

Based on the conducted experiments in simulation it becomes evident that all maps behave as designed with input data mimicking the expected real world data. Expected performance impacts resulting from obstacle shadowing and the oblique angle and proximity of the camera and the ground are measurable. None of the observed edge cases occurred in plain sight of the navigation cameras or posed a drive hazard for the rover.

The main observed disadvantage of piecewise constant maps for the given configuration lies in their inability of resolving any size of obstacles at the low resolutions. Additionally, obstacles detected at a distance on medium resolution layers might disappear on these layers as soon as the rover comes too close to them.

These issues are not present when using the CCM update. Even subcellsize obstacles are reliably detected. The additionally gained DEM precision per used memory and cell is an additional strong point for

Figure 7.52: Photo of the Test Scene.

CCM. The major observed drawback of CCM is the minimal spatial resolution of 4 cm. In certain circumstances, larger terrain patches are classified false negative compared with piecewise constant maps. CCM requires more tuning on the real hardware to avoid false negative classification of the terrain.

7.5 EVAL ON MERCURY 7

The evaluation on real world data was carried out similarly to the previously described simulation based evaluations. Instead of using an obstacle size based metric, only the frame by frame analysis yields usefull. Due to significant shadowing of medium to large size rocks and the limited available space, not enough obstacles of different sizes are present in order to perform the evaluation in the obstacle size domain.

This evaluation is still providing insight into the performance of OMG and CCM obstacle detection performance. It essentially forms the validation of the previous evaluations on synthetic data.

7.5.1 *Real World Setup*

For the real world evaluation a test scenario at JPLs Mini Mars Yard was set up. On a region of 5×5 m rocks of different sizes and one small (traversable) crater were placed.

A Vicon setup with 7 Vicon Vantage cameras is used for external pose reference. Due to the unavailability of the finished localization algorithms at the time of testing, this external reference system was used for mapping.

Figure 7.52 shows a photo of the test setup.

During data capture, the rover was driven manually. Trajectories were selected in order to cover as much of the drivable area as possible without leaving the Vicon covered region.

7.5.2 *Long Range Data Issues*

It is important to mention, that in particular the coarse low resolution maps are non evaluatable with this setup. The drivable test area is not big enough to traverse enough distance for the low resolution layers to make sense. Additionally, JPL infrastructure surrounding the test area causes stereo noise which propagates into the evaluation area. Therefore, the low resolution layer evaluation is suggested as a future validation test.

7.5.3 *Piece Wise Constant*

The first run is a simple and short trajectory resembling a nominal drive scenario.

The same frame by frame analysis as previously done for simulation is performed on real data.

The analysis in Figure 7.53 show the expected bad performance for level 3 and 2. No obstacles were present which could have triggered the roughness filter for level 3. The aforementioned wall issue would also make bigger obstacles nearly impossible to detect without the presence of a major slope.

Looking at level 1 a steady false positive count is visible over the entire length of the data set. Figure 7.54 shows the map plot at T=200 s. The prominently not detected obstacle has a height of 6 cm and therefore cannot be detected at this level. Two other obstacles of same size are causing a similar issue of which one is always visible to the map at a time. This type of false positive is therefore not concerning. No obstacle of detectable size remained undiscovered. On the highest resolution (level 0) all obstacles are picked up as shown in Figure 7.55.

The apparent slight underestimation of the obstacle size is an effect of how the ground truth obstacle map was created. Similar to the picewise constant traversability analysis a roughness filter is used to label the hazardous areas. This overestimates the obstacle size by approximately the size of the roughness filter itself. The true obstacle is therefore slightly smaller and completely covered by the OMG estimate.

On some instances during the run the map shows seemingly arbitrary false negative patches as for example at T=100 s displayed in Figure 7.56. Those false negatives are mostly disturbances in the sand. Since the roughness filter of the mapping must be tuned more aggressive than the one used for ground truth generation, those *potholes* do not show up in the ground truth hazard map. A false negative producing example is shown in Figure 7.57. The diameter and depth are too small to cause

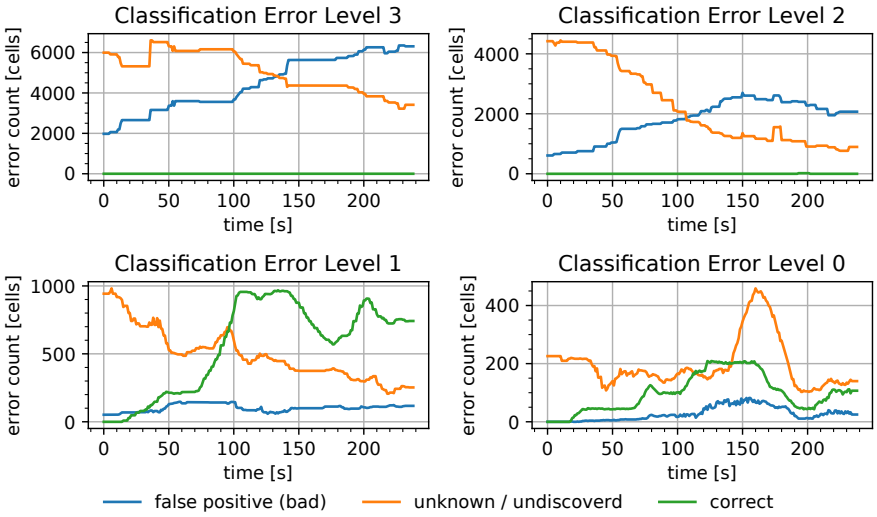


Figure 7.53: Real World Scenario 1 Frame By Frame Analysis OMG.

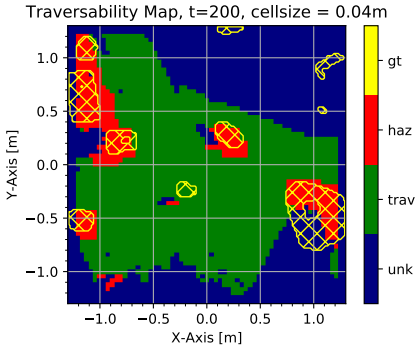


Figure 7.54: OMG Map Plot at T=200 s, Layer 1.

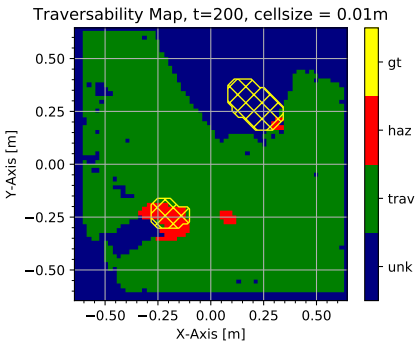


Figure 7.55: OMG Map Plot at T=200 s, Layer 0.

an actual drive hazard. Their identification therefore is erroneous. In other instances, small false negative areas are caused simply by a less observed (shadowed) region being influenced by stereo noise.

Especially waves in the sand or small rocks, that shadow some cells were observed to cause this issue.

The visible false positives around 150 s on Layer 0 are caused by exactly those effects and do not resemble a drive hazard.

7.5.4 *CCM*

Again, we start with an analysis of the frame by frame metrics in Figure 7.58.

With a side length of 6 m and 10 m level 4 and 3 are not properly evaluatable with the used setup.

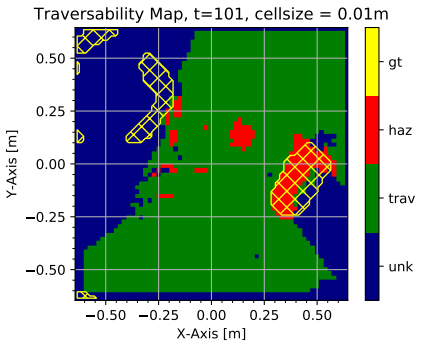


Figure 7.56: OMG Map Plot at T=101 s, Layer 0.

Figure 7.57: Disturbance in the Sand Causing False Negative

Level 2, 1 and 0 show overall low false positive rates. Since this evaluation does include all ground truth obstacles on all layers, a closer look into the individual maps is needed to get an impression about the performance.

Starting with level 2, a small but over the entire run present count of false positive classifications is visible.

Exemplarily evaluated at T=200 s mainly two rocks which are in range of the map for the entire time contribute to the false positive rate. The associated snapshot is shown in Figure 7.59 Both rocks have a height of 6 cm and are therefore undetectable in level 2. The same timestamp is also plotted for level 1 in Figure 7.60. Here, the 6 cm rocks are within the detection margin and are correctly classified as obstacles.

Over the entire run, level 2 shows significant amounts of false negative detections. The main reason for that is again stray measurements from stereo noise. Also shadowing and increased stereo variance at obstacle borders is a big contributer to those false negative detections. If no obstacles are close, also the low resolution levels correctly identify the area as traversable. A snapshot of increased false negative detections on level 2 is shown in Figure 7.61.

Level 0 unsurprisingly shows the highest precision of all. Figure 7.62 plots the same timestamp as before (T=200 s) but on level 0. An excellent classification is visible. The slight visible underestimation is again caused by the real world over estimation of obstacle size, as discussed earlier.

Level 1 performs well.

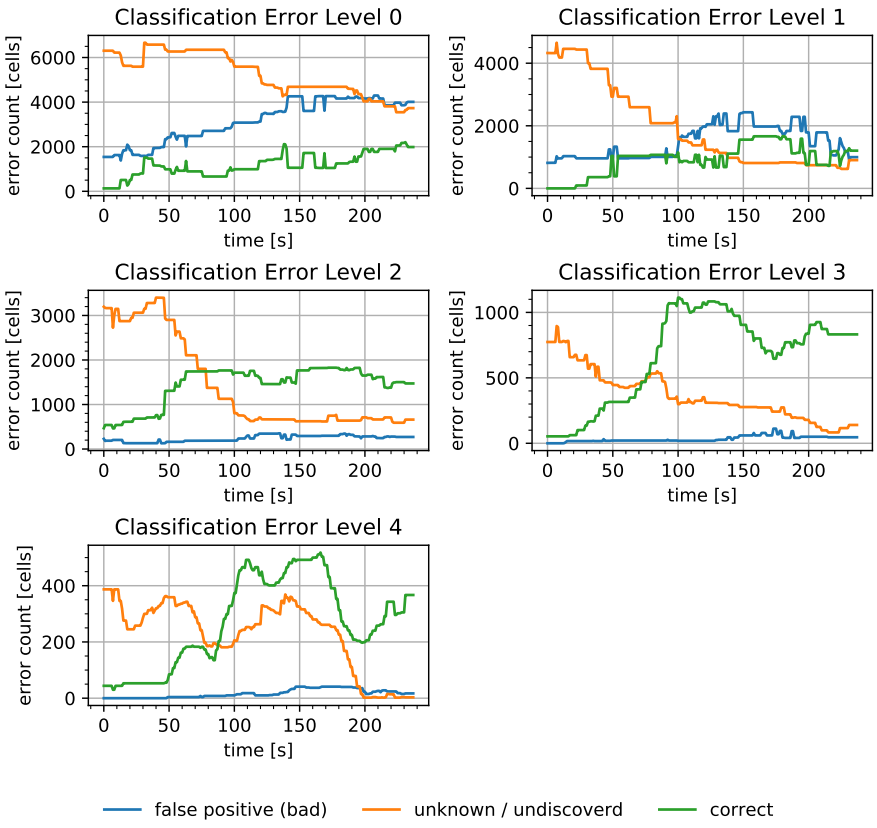


Figure 7.58: Real World Scenario 1 Frame By Frame Analysis CCM.

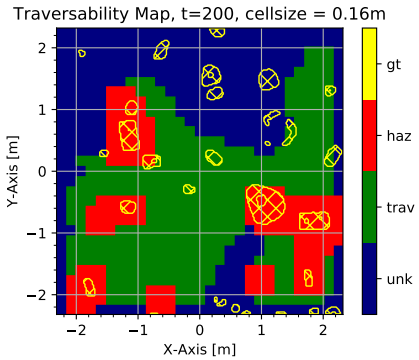


Figure 7.59: CCM Map Plot at T=200 s, Layer 2.

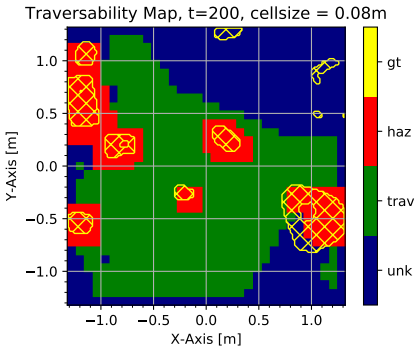


Figure 7.60: CCM Map Plot at T=200 s, Layer 1.

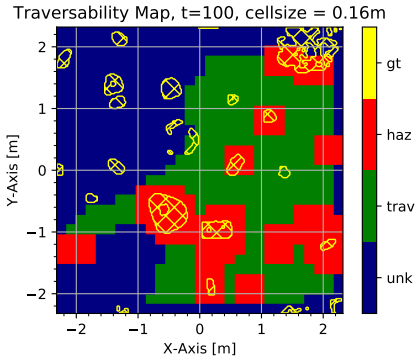


Figure 7.61: CCM Map Plot at T=100 s, Layer 2.

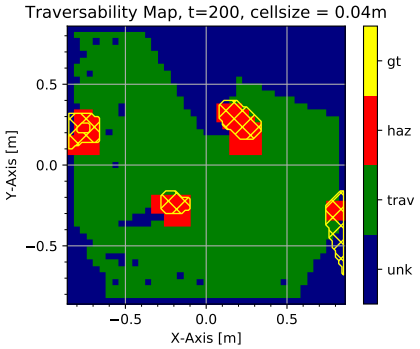


Figure 7.62: CCM Map Plot at T=200 s, Layer 0.

7.5.5 *Height Precision Evaluation*

To compare the precision of the mapping approaches in the real world level 1 was selected for piece wise constant and also level 1 for CCM. These levels are small enough to avoid obstacles which are not part of the test area while having a coarse enough resolution to cover a significant ground truth area. Ground truth is evaluated at a spacing of 1 cm. Table 7.6 summarizes the parameters of the two chosen layers. Due to the slope consideration of CCM they differ in resolution and cell count. They are comparable in terms of expected performance, size and memory usage.

Parameter Comparison		
Element	Piece Wise Constant	CCM
Size	260 cm	264 cm
Resolution	4 cm	8 cm
Number Cells	4225	1089
Smallest Detectable Obstacle	7 cm	6 cm

Table 7.6: Parameter Comparison

To benchmark the map precision a dataset section has been chosen that covers a crater and multiple small obstacles. The rover drives through the crater and sees some small obstacles on the other side. Figure 7.63 shows the ground truth terrain of the selected area as a point cloud. The currently by CCM covered area marks a snapshot in the middle of the 30 s dataset. Having a crater in the dataset proposes an additional challenge due to areas of high inclination and initially low coverage.

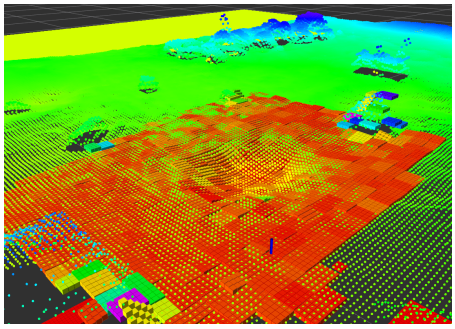


Figure 7.63: Selected Terrain for Precision Evaluation

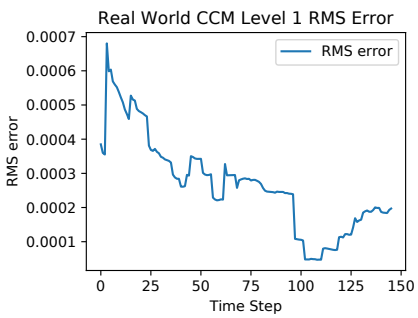


Figure 7.64: CCM RMS Precision over Time.

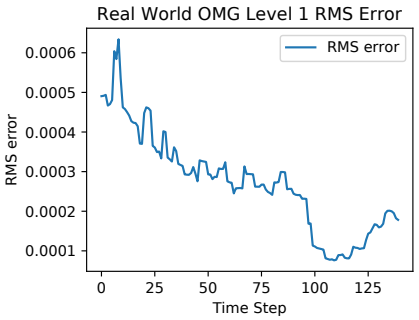


Figure 7.65: OMG RMS Precision over Time.

7.5.5.1 *RMS Error Over Time*

Figure 7.64 and 7.65 show the RMS height precision evaluation result for the selected layers. Despite the difference in resolution, both maps have a very close RMS error and therefore represent the ground truth surface equally well. This result matches our preliminary evaluations based on simulation from Section 6.6. The presence of obstacles degrade the analysis results. Obstacles are in general hard to represent in detail by all maps. Therefore, the total RMS error is higher in this analysis, but relative to each other still valid.

7.5.5.2 *Variance and Error Map*

Having a closer look to the differences posed in the variance representation more of the initial claims become evident. Both height maps look unsurprisingly quite similar as visible in Figure 7.66 and 7.67. The variance maps on the other hand are significantly lower and more concentrated around obstacles for CCM compared with OMG. Additionally, CCM has significantly fewer variance violations, which makes the slope normal variance estimation of CCM more precise and reliable compared with OMG. If the feature of the true surface laying within 3σ of the estimated height is critical for one's application CCM has a

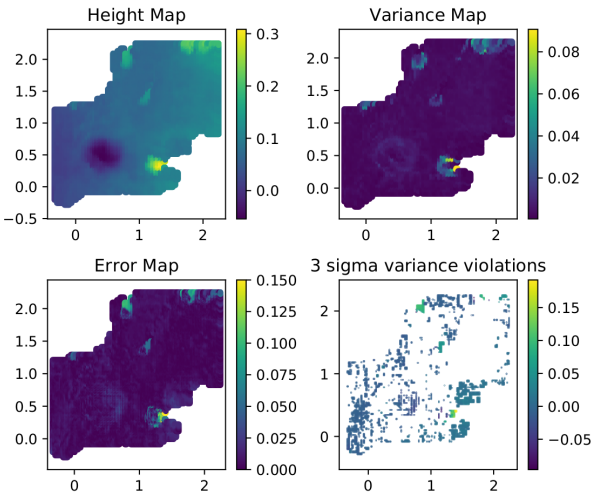


Figure 7.66: OMG Height Map, Variance Error and Variance Violations.

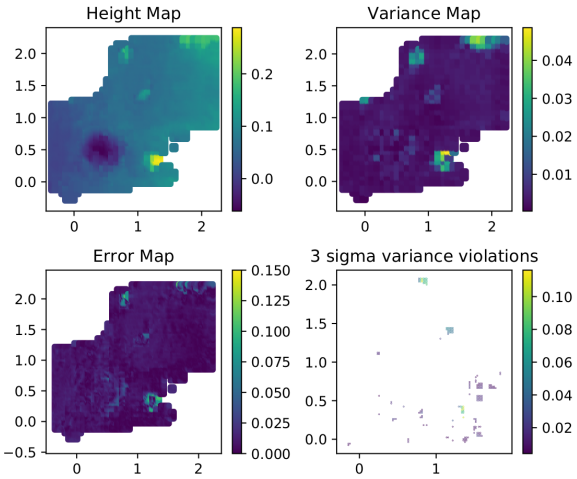


Figure 7.67: CCM Height Map, Variance Error and Variance Violations.

clear advantage. These results were validated on multiple datasets at all time stamps.

7.5.6 *Real World Conclusion*

Despite an expected increase in false negative classifications, no new effects or corner cases were observed in the real world campaign. The real world results closely match the from simulation expected behavior. No parameter tuning or other adaptions were necessary from theoretical values to simulation and finally the real rovers. Obstacle detection performance matches the expectations.

Further analysis for the low resolution layers and crater shaped obstacles should be performed to properly cover the operation space of the map with real world evaluations.

7.6 DISCUSSION

7.6.1 *Memory and Runtime*

Part of the present evaluation is the consideration of memory usage and runtime on target hardware for the presented techniques and selected parameters.

7.6.1.1 *Memory Usage*

Table 7.7 shows the memory usage of CCM per layer and in total. All memory values are given in *numbers of used floats*. The alternate stack up shows an optional layer selection based on the most useful sizes. A single CCM cell has nine float values while OMG needs three and Kalman just two.

Table 7.8 is a similar setup but for the piece wise constant updates OMG and Kalman.

Memory Usage CCM			
Layer	Size	Memory All	Memory Alternate
0	17	2,601	2,601
1	19	3,249	-
2	29	7,569	7,569
3	33	9,801	-
4	43	16,641	16,641
Total		39,861	26,811

Table 7.7: CCM Memory Usage in Number of Floats

Memory Usage PWC			
Layer	Size	Kalman	OMG
0	17	578	867
1	33	2,178	3,267
2	65	8,450	12675
3	129	33,282	49,923
Total		44,488	66,732

Table 7.8: PWC Memory Usage in Number of Floats for OMG and Kalman Updates.

8

FLIGHT SETUP, CONCLUSION, OUTLOOK AND FUTURE RESEARCH

This thesis presents a mapping and hazard detection pipeline tailored to the requirements of the CADRE mission. The presented technique based on two different mapping algorithms efficiently detects drive hazards and provides a high resolution DEM for global mapping. This chapter discusses the final map setup and the next necessary steps for the presented technique to be used on the moon. Concluding the previous chapters, final design decisions and trade off selections are outlined. A section covering possible future areas of research and applications wraps up this work.

8.1 FLIGHT SETUP

The analysis presented in this work considers different mapping techniques individually. Scientifically, perks and drawbacks of individual approaches are determined and discussed.

From an engineering point of view, benefits of different mapping updates can be combined for an optimal result. Changing the map update for individual layers enables the use of suitable techniques for different resolutions. The algorithm decision is based on the extensive testing described in previous chapters.

8.1.1 *Mapping*

Based on the evaluations presented in this work, a combination of the Kalman mapping update and the new CCM algorithm is chosen for the final flight map.

This heterogeneous setup combines the strengths from both techniques. As a conservative, well analyzed algorithm, the Kalman map does not have many tunable parameters, which need to be adapted to the conditions on the moon. Due to the numerous drawbacks described in previous chapters, it is only used for the highest Layer at 1 cm resolution. This layer must be the most reliable, since it is primarily used for hazard avoidance by the local planner. The high resolution obstacle map is beneficial for the local planner and allows precise planning. Sub cell slope information is not needed on a 1 cm grid and no drive hazards

with a footprint smaller than 1 cm are expected. Kalman saves memory and is therefore the optimal choice for the dimensionally limited first layer.

All other layers use the CCM update to estimate the full covariance representation. The used layer sizes for CCM are `[4, 16, 64]cm` and cover the whole 10x10m on the lowest resolution. Both, the 1 cm and 4 cm layers are capable of resolving the minimal obstacle size. Ahead planning is enhanced by the better obstacle detection performance, while the terrain reconstruction precision is also improved.

The low resolutions are mainly used for map fusion, global planning and far-ahead drive planning. Measurement sparsity resilience and enhanced obstacle detection at distance justify the more complicated algorithm for these layers.

This heterogeneous map setup is integrated into the existing framework.

8.1.2 *Implementation*

On the implementation side, only few changes were made to the previously described setup. Most prominently, map pooling is replaced with more simple all-layer insertion of measurements. Since layer 0 runs a different update algorithm then the other ones, pooling would only have been possible for the three CCM layers. A runtime increase by a factor of 1.8 from pooling to all layer insertion does not justify the added complexity in software and testing. Runtimes are still in the acceptable range.

A new map class combining the map updates in the presented way replaces the different map update classes and forms the final flight map. Unnecessary functions and evaluation methods simply got cleaned up and the ROS wrapper is replaced with an F′ one.

This completes the changes necessary for the flight software implementation.

8.2 CONCLUSION

The work described in this thesis builds upon an extensive literature survey on existing mapping techniques for mobile robotics. Multiple state-of-the-art mapping algorithms and frameworks were implemented and evaluated in context of the CADRE mission. A new highly efficient, robot-centric mapping framework based on rolling buffers has been developed. Multiple map layers of different resolution, size and parameter count are supported. Position based indexing enables individual

layer map rolling and layer pooling for highly efficient incremental measurement updates.

The framework is implemented in C++ without the use of any external libraries or dynamic memory in compliance with NASA flight software coding standards. Attention was paid to a fast and robust execution of the code on low performance embedded processors. The finalized software does not rely on external implementations and therefore can run by its own. Individual methods were continuously checked by unit testing, while the setup as a whole is checked and evaluated in multiple simulation and real world based test campaigns.

For further evaluation, two state-of-the-art maps (Kalman, OMG) were implemented using the new framework. Additionally, a new, application tailored, covariance based map was developed and applied. All three maps are extensively evaluated in simulation and real world test campaigns. Ultimately, the selected CADRE flight mapping setup is permanently integrated into the framework and handed over for further V&V testing.

The new CCM mapping algorithm presented in this work is based on the known idea of estimating the measurement sample distribution within individual cells. An incremental update enables efficient insertion of new measurements. CCM is capable of merging multiple high resolution cells into one low resolution cell for multi resolution maps. The covariance based algorithm is capable of representing terrain to the same precision of a conventional piece wise constant map of double the resolution. Estimation of surface-normal sample variance allows reliable detection of sub-cellsized traversability obstacles. Map design parameters and considerations were outlined on the example of the CADRE rover.

Using the CCM algorithm comes at the cost of slightly increased computational and memory usage for an obstacle map at the same resolution. The slope estimation makes this map update highly sensitive to sensor and pose noise. Therefore, a correct selection of map resolution with consideration of the measurement variance is essential. Especially, obstacle detection parameters are less intuitive to select and tune compared with conventional techniques. More setup specific parameters effect the convergence and ultimately usability of the resulting map.

Concluding, the new technique brings significant improvements and is successfully applied in the CADRE mission.

8.3 OUTLOOK

With a functional and guideline conform map on hand, more verification and validation testing needs to be done. As soon as the full navigation stack is ready, the map will be evaluated in more real world testing campaigns. The full map footprint on all layers will be tested for artifacts and obstacle performance on rocks and craters. If necessary, small adaptions for runtime and memory usage can get applied.

These tests target the verification of the full software stack. Mapping is performed with the pose determined by the flight software state estimator. Degrading effects from real world pose drift and noise are captured this way. Similar to the real mission, multiple rovers operate simultaneously to generate *dynamic* obstacles.

Later in 2023, the software will be finalized and handed over for integration on the flight hardware. At the time of writing, a launch date of the CADRE mission between January and June 2024 is planned.

- GO CADRE!

8.3.1 *CCM additions*

In its current state, CCM is only used for surface reconstruction. As described by [69] covariance based maps can be used for more efficient global alignment. Currently, the global map aligns the input maps by super sampling on the slopes and subsequent application of ICP. A smart usage of the multiple resolutions and full 3D covariance could result in a much faster and more precise alignment of sub maps.

By estimating the RMS error for each insertion, a similar technique could be used for faster VIO based on a known or partially converged map.

8.4 MARS SAMPLE RETURN

One of the major weak points of the newly presented CCM algorithm is the sensitivity to partial observation. In a rover setup, rough terrain with features of similar size to the map resolution causes partially observed cells. Those cells cannot be evaluated for slope which makes the map prone to false negative detection and decreases the precision significantly. When observing a scene from above, all cells can be observed without issues caused by shadowing or decreasing point density. Additionally, measurements are distributed uniformly which simplifies slope estimation.

Thus, using the CCM map on an areal platform for mapping and landing site detection is a predestined application. Future Mars missions such as Sample Return Helicopter (SRH) as part of the Mars Sample Return Mission or Mars Science Helicopter (MSH) [85] will incorporate aerial vehicles similar to the wildly successful Ingenuity helicopter. Especially for the Mars Sample Return mission, a great amount of autonomy and reliability is required for a successful completion. The Sample Return Helicopters have to navigate autonomously and in case of an emergency, abort to the last known safe landing site.

The by then flight proven CCM algorithm is an excellent candidate for this task. Detecting sub cell size landing hazards can be a significant improvement over existing landing site detection algorithms.

8.5 FUTURE RESEARCH

An obvious limitation of the presented mapping framework is the constraint to 2.5D elevation maps. Any robotic system operating in enclosed spaces is not supported by the current setup. Extending the current data structure to an oct-tree based structure would support covariance surfel estimation at arbitrary positions. This setup would still allow a multi resolution multi size map in 3D.

Especially earth bound mobile robotics face the challenge of encountering obstacles of different severeness. While for some robotic platforms soft obstacles such as high grass does not pose a thread, pure geometry based mapping is unable to distinguish between traversable and non-traversable obstacles. To overcome this limitation, semantic labels can be attached to each cell. The *true* hard surface beneath could be estimated as a second map layer based on surrounding data. Capable robotic platforms would therefore be able to traverse soft obstacles.

The current planner uses a trinary traversability map. Based on the measured cell roughness, confidence and slope, a traversability score could be formulated. This would allow a more advanced planner to select a path based on acceptable risk. The traversability score could be augmented with system related features such as shadowing of solar panels, orientation of science instruments or similar platform dependent parameters.

A

GROUND TRUTH FROM LASER SCANS

A.1 USE LASERSCANS AS GROUND TRUTH FOR REAL EVAL

For the Experiment Ground Truth generation a Leica BLK 360 laser scanner was used. Unfortunately, the BLK only outputs data in the proprietary format `.blk`. The `.blk` format has no documentation, is non human-readable and contains more than just the point cloud. Conversion of this format to something usable can only be done in two ways:

1.
   - Take scans manually by pushing the button on the scanner.
   - Transfer the scans to a Windows computer with the free BLK360 Data Manager.
   - Use Leica Cyclone Register to convert the point clounds. This software is costly

2.
   - Use the IPad App Leica BLK360 to take the scans.
   - Transfer the scans to the IPad by creating a new project and importing the scans as scenes.
   - Export each scan individually as E58 to the IPads' internal storage.
   - Transfer the E58 files from the IPad to the workstation of choice.

   Please note, that the other offered apps (BLK Live) do NOT work. Leica Cyclone Field also requires a paid license.

A.2 POST PROCESS POINT CLOUDS AS GROUND TRUTH DATA

In this section the ground truth data post-processing is described. The post processing workflow requires multiple steps:

- Data Conversion `.e57` →`.xyz`
- Alignment of the Scans
- Trimming of the Combined Point Cloud

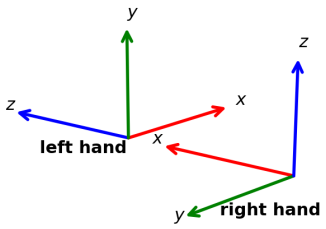


Figure a.1: 3DTK's Left-Hand and Usually Used Right-Hand System

- Finding a Transformation from Vicon Frame to Pointcloud Frame
- Generating Ground Truth Data Which can be Used for Evaluation

A.2.1 *Data Conversion*

Data conversion from `.e57` to `.xyz` has to be done two times in the process. Once right after acquisition and once again after trimming. The most convenient tool for the job is Meshlab. Simply import the `.e57` and save as `.xyz`. Make sure to un-tick the *normal* checkbox during export.

A.2.2 *Alignment of the Scans*

If available, initial alignment can be done with the Leica Cyclone Register software. A free option would be Cloud Compare, which we will later use for trimming the point cloud. My option of choice is 3DTK, a command-line point cloud toolbox with highly precise point cloud registration.

To import the scans in 3DTK all `.xyz` files have to be within one folder. They must be renamed to `scan000.xyz`, `scan001.xyz`... Additionally, each scan must be associated with a `.pose` file as in `scan000.pose`. This file is used for rough manual alignment to make the job of the registration algorithm easier. Initially each `.pose` file has the following content:

```
0 0 0
0 0 0
```

The first three values represent translation in x, y, z, while the second three values are rotation around x, y, z. Rotations are expressed in degree. A left-handed coordinate system is used for alignment. The used coordinate system is shown in Figure Referencesfig:laser/leftToRight.

At this point, the working directory should look like this:

```
.
├── scan000.pose
├── scan000.xyz
├── scan001.pose
├── scan001.xyz
├── scan002.pose
├── scan002.xyz
├── scan003.pose
├── scan003.xyz
```

With everything in place, we can dare a first look at `scan000.xyz` with the following command:

```
    slam6d-code/bin/show pwd_to_working_dir -s 0 -e 0 -f xyz --
autoOct --advanced
```

For a detailed explanation of the used arguments please look into the help page by using `--help`. We can navigate the camera by pressing `[w, a, s, d, y, c]` for translation, `[q, e]` for rolling and `[i, j, k, l]` for pivoting. It is recommended to enable point cloud reduction for better translation performance.

In most of the cases the ICP algorithm is capable of finding the correct correspondences without manual pre-alignment. The following two sections describe the process with and without manual pre alignment. Manual alignment is usually only necessary when little to no big features are present in the point cloud. The actions described in the following section are only relevant if little to no big features are present in the point cloud. Usually one can skip directly to using the ICP algorithm as shown in a.2.2.2.

A.2.2.1 *manual pre alignment*

Before handing the heavy work over to the registration algorithm, we have to align the point clouds corse. For this task, we first have to show both, the base point cloud (000) and the first point cloud (001).

```
[bash]
    slam6d-code/bin/show pwd_to_working_dir -s 0 -e 1 -f xyz --
autoOct --advanced
```

In the right-hand menu under `Color: Color type:` one can select `ID Scans by Color`. This makes alignment much easier. Open `scan001.pose` and guess values for translation and rotation. After modification and saving we have to update the frames files from the new pose. This is done by :

```
[bash]
    slam6d-code/bin/pose2frames pwd_to_working_dir
```

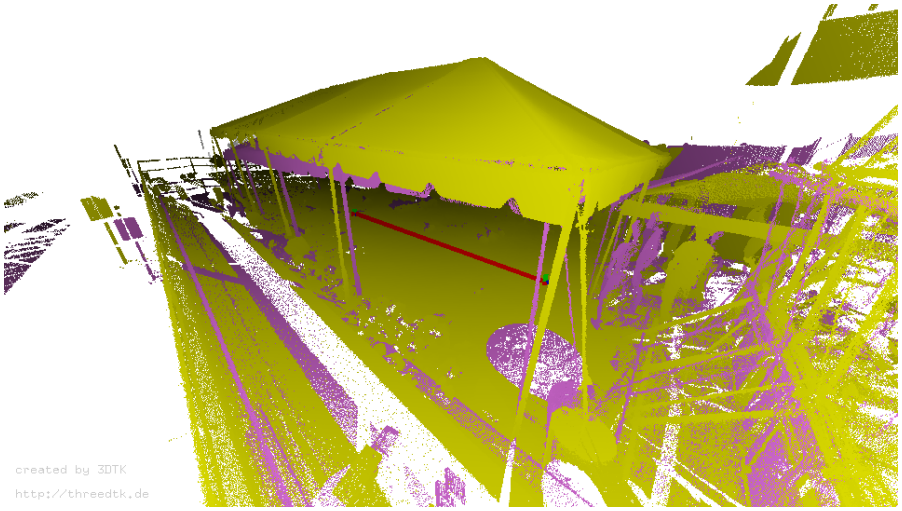


Figure a.2: Rough Point Cloud Alignment

To refresh the view in show simply use the button `advanced/Relaod frames`. This process must be repeated until both scans are roughly aligned. A rough alignment as shown in Figure Referencesfig:laser/roughPCalign completely suffices at this point.

All other scans must be aligned the same way. The `-s` and `-s` parameters from `bin/show` must be adjusted to show the corresponding point clouds. Sorry, this task is a little tedious and takes a while.

A.2.2.2 *ICP alignment*

For the ICP alignment 3DTKs' slam6d tool is used. Usage of this tool can be a bit tricky. Therefore I strongly suggest reading the help page and playing around with the tool. To reduce computation time it is a smart move to start with a high level of point cloud reduction. After the initial alignments, finer alignments can be made until a satisfying result is achieved. Typically, my first run of slam6d looks like that:

```
    bin/slam6D pwd_to_working_dir -s 0 -e 6 -f xyz -i 500 -r 50 -
d 200
```

This takes all (6) scans into consideration, sets a maximum iteration count of 50. The `-r 10` preprocesses the point clouds to have a resolution of 10 cm. `-d 100` sets the maximum correspondence distance to 200 cm

After the first run, *slam6D* has crated `.frames` files for each scan. To use those transforms for the next runs start, the transformations need to be moved to the `.pose` files. The following tool updates the `.pose` files.

```
[bash]
    slam6d-code/bin/frames2pose pwd_to_working_dir
```

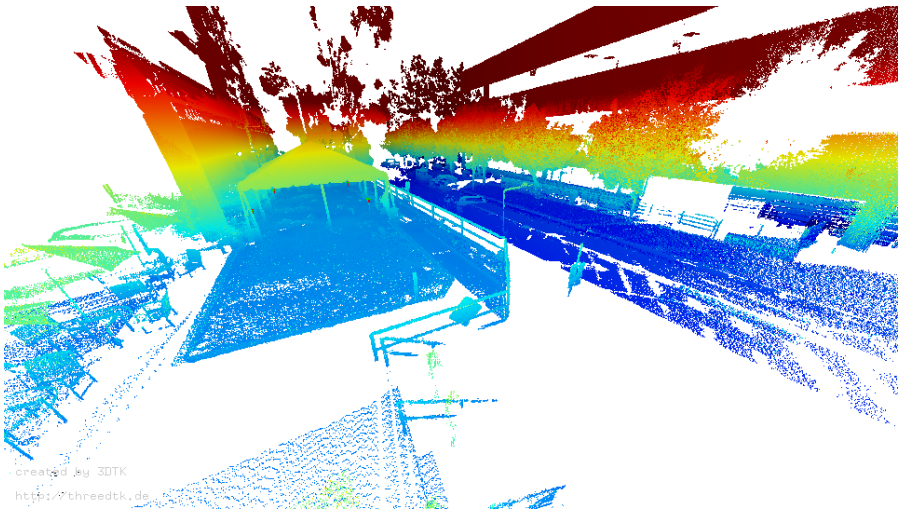


Figure a.3: Final Alignment

After updating the `.pose` files, the same process has to be re-done. I received good results by running the following pointcloud reduction steps. The correspondence distance can be reduced as well.

- 50
- 20
- 10
- 5
- 2
- raw (no `-r`)

After checking the results with `show` finer resolutions can be selected. As soon as one is happy with the achieved accuracy, the combined point cloud can be exported with:

```
    slam6d-code/bin/exportPoints pwd_to_working_dir -s 0 -e 6 -
f xyz -x
```

A `points.pts` file containing all points combined is generated in the calling directory.

This completes the point cloud alignment procedure. The final result can look like Figure a.3.

A.2.3 *Trimming of the Combined Point Cloud*

In some cases, there might be unwanted objects within the laser scan, such as roofs. Also surrounding items, which are not part of the region

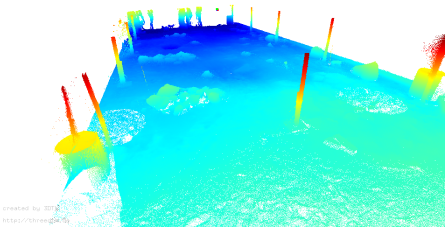


Figure a.4: Trimmed Point Cloud of Mini Marsyard.

of interest should be excluded from the final data product. For trimming the point cloud *Cloudcompare* is a good option. It offers fast rendering of large amounts of data while having a good interface for trimming the point cloud. Please make sure to install a version > 2.11 (apt version is older at the time of writing this report (09/03/2022)). Older versions do not have the ability to export `.e57` files, which we need for further processing. The version within Ubuntu's snap store works.

Trimming the point cloud is as easy as using the segmentation tool in Cloudcompare and exporting the remaining parts. After exporting, the point cloud should be converted back to `.xyz` data by using e.g. Meshlab.

The trimmed example point cloud of the Mini Marsyard now looks like Figure Referencesfig:laser/cutPointCloud.

A.2.4 *Finding a Transformation from Vicon Frame to Point Cloud Frame*

This part of the evaluation data pre-processing requires the most custom software. It is especially important to have the ground truth odometry data in the same coordinate frame as the ground truth map data. Only when this is given, a qualitative analysis of the mapping height map output is possible. To achieve such correspondence at least three points need to be known in both, the Vicon frame and in the point cloud. Even though laser scanning is a highly precise process compared to stereo camera based slam, it is not sufficient to simply select a point in the point cloud and use it as a reference. A target the size of a Vicon marker is also not guaranteed to be visible in the laser scan.

The proposed setup first gathers the position of at least three Vicon markers by the Vicon system. Those markers shall be distributed over the entire drivable area. After acquisition of the markers, 70 mm laser sphere targets are placed on top of the Vicon markers. The assembly shown in Figure a.5 ensures that the center of the Vicon marker is also the center of the laser sphere. With this setup in place, the laser scans are taken.

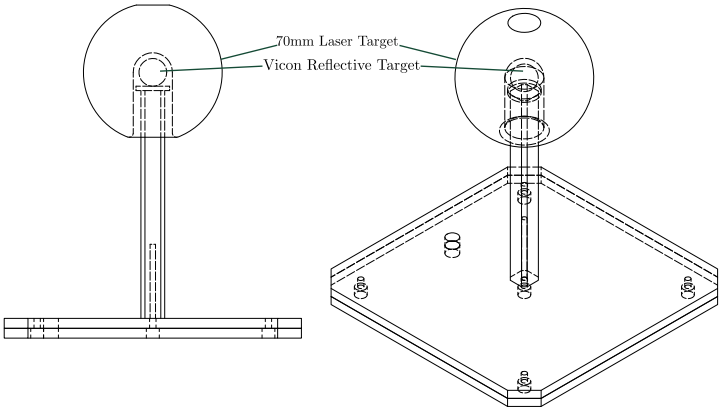


Figure a.5: Drawing of the Alignment Spheres. Outer Shell poses a Laser Target and Covers the Inner Reflective Target for the Vicon System.

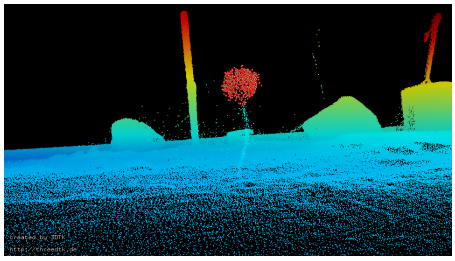


Figure a.6: Laser Target Selection.

After alignment and trimming of the point clouds the points associated with each sphere must be extracted to get their exact position in the point cloud frame. Again, 3DTK's show tool is used for that. Similar to the individual scans, the combined point cloud is loaded in 3DTK. Navigate to one of the alignment spheres and untick the `mouseNav` checkbox in the horizontal menu. This way point selection is activated. Select the points and export them. The selection should look something like Figure a.6.

Attention! The generated `.3d` files are in the left-handed coordinate system as previously shown in Figure a.1. A python script is now used to fit a sphere to the extracted point cloud. This way, it can be ensured that the center of the sphere and therefore the alignment point is found even if we experience unequal or partial coverage on the target. The provided sphere fitting script already takes care of the left-handed coordinate system. Any output will be in the usual right-handed coordinate system. Fitting a sphere to the extracted data will look like Figure Referencesfig:laser/PointsAndFittedSphere

The way more interesting output is the fitted sphere diameter and center coordinates. For this example the output is: `diameter: 67.41mm, X: -842.49cm, Y: 349.62cm, Z: -28.63cm` The diameter output can be

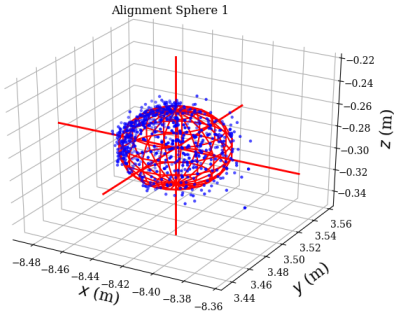


Figure a.7: Extracted Points and Fitted Sphere.

used to check if something went wrong. It should be close to the actual size of the used laser target spheres.

The same python script also uses previously captured Vicon data to calculate the Transform for the given point cloud. For doing the entire process, the selected spheres shall be stored in the same directory as the combined point cloud `scan000.xyz`. The sphere selections must be named `sphere0.3d`, `sphere1.3d`...Additionally, a file called `vicon.csv` must contain the Vicon position measurements of the alignment spheres. It's content looks like

```
x, y, z
0.000000, 0.000000, 0.000000
0.000000, 7.816153, 0.000000
-2.423199, 6.974209, 0.000000
```

It is important to keep the correspondences in mind, when naming the spheres file. `sphere0.3d` corresponds to the first entry in the `vicon.csv` file and so on.

With everything in place, the `alignPCwithVicon.ipynb` notebook can be executed. It will start with fitting spheres to the extracted point clouds and outputting the results. Following, the Vicon coordinates are loaded, and dimensional errors are calculated. Those errors are an indicator, if the overall setup is sound and correct correspondences were selected.

After calculating the optimal transformation from the Vicon frame to the point cloud frame, the original point cloud `scan000.xyz` is loaded, transformed with the calculated transformation and stored as `transformed.xyz` within the same directory.

It is recommended to copy this file to a new folder, rename it to `scan000.xyz` and add a `scan000.pose` file to inspect it with show. This step is also necessary in order to proceed with the ground truth reference gen-

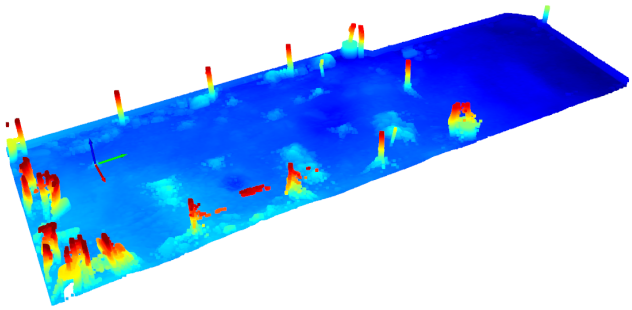


Figure a.8: Vicon Frame Aligned Point Cloud With Coordinate System.

eration. The new point cloud now aligns with the Vicon frame. As a reference, the aligned example cloud is shown in Figure a.8

A.2.5 *GT Reference Generation*

As a last step before running the evaluation, we must convert the aligned point cloud to the ground truth format. The ground truth format used by the live evaluation node is `.jdem` (JPL Digital Elevation Model). `.jdem` Files are a discretized version of the real terrain (2.5D) height or hazard map. Internally, the data is stored in a row - major fashion and meta-data is stored in a human-readable header.

As parameters, the working directory, ground truth resolution and side length must be specified. An exemplary call would look like:

```
rosrun utilities pc2jdem _working_dir:=/pwd/to/directory
_resolution:=0.01 _side_length:=20
```

`pc2jdem` centers the output around the mean of the point cloud. Offset values are set in the header to keep data aligned with the Vicon frame.

Figure a.9 shows the drivable area of the generated DEM. A second python script filters this DEM for obstacles. Obstacles are stored in the same `.jdem` architecture in a different file. The pixel value is the region height of the obstacle. This way, the evaluation algorithm knows the size of the obstacle for classification. Figure a.10 shows the hazard ground truth. All non-zero values are associated with a hazard.

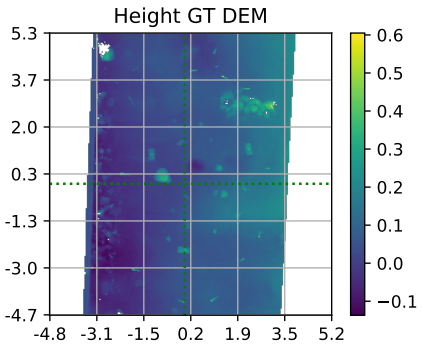


Figure a.9: Ground Truth DEM from Laser Scan

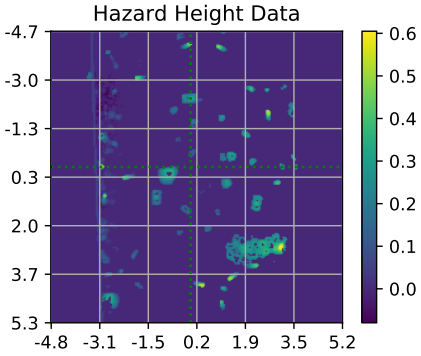


Figure a.10: Hazard Map from Laser Scan

BIBLIOGRAPHY

[1] Issa A.D. Nesnas, Lorraine M. Fesq, and Richard A. Volpe. "Autonomy for Space Robots: Past, Present, and Future." en. In: *Current Robotics Reports* 2.3 (Sept. 2021), pp. 251–263. ISSN: 2662-4087. DOI: 10.1007/s43154-021-00057-2.

[2] G. Rabideau, V. Wong, D. Gaines, J. Agrawal, S. Chien, E. Fosse, and J. Biehl. "Onboard Automated Scheduling for the Mars 2020 Rover." en$_U$S. In: (Oct. 2020). Accepted: 2022-01-11T20:54:29Z. URL: https://trs.jpl.nasa.gov/handle/2014/53238.

[3] Neil Abcouwer, Shreyansh Daftry, Tyler del Sesto, Olivier Toupet, Masahiro Ono, Siddarth Venkatraman, Ravi Lanka, Jialin Song, and Yisong Yue. "Machine Learning Based Path Planning for Improved Rover Navigation." In: *2021 IEEE Aerospace Conference (50100)*. Mar. 2021, pp. 1–9. DOI: 10.1109/AERO50100.2021.9438337.

[4] JPL. "JET PROPULSION LABORATORY 2019 STRATEGIC TECHNOLOGIES." en. In: (), p. 76. URL: https://scienceandtechnology.jpl.nasa.gov/sites/default/files/documents/JPL_Strategic_Technologies_2019.pdf.

[5] https://www.jpl.nasa.gov. *CADRE Project*. en-US. Dec. 2022. URL: https://www.nasa.gov/directorates/spacetech/game_changing_development/projects/CADRE.

[6] Dec. 2022. URL: https://www.nasa.gov/feature/npr-71208-revision-a-nasa-research-and-technology-program-and-project-management.

[7] Dec. 2022. URL: https://www.nasa.gov/content/commercial-lunar-payload-services.

[8] M. Kurata, H. Tsunakawa, Y. Saito, H. Shibuya, M. Matsushima, and H. Shimizu. "Mini-magnetosphere over the Reiner Gamma magnetic anomaly region on the Moon." en. In: *Geophysical Research Letters* 32.24 (2005). ISSN: 1944-8007. DOI: 10.1029/2005GL024097. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2005GL024097.

[9] https://www.jpl.nasa.gov. *A CADRE of Mini-Rovers Navigate the Lunar Terrain of SLOPE*. en-US. Dec. 2022. URL: https://www.jpl.nasa.gov/news/a-cadre-of-mini-rovers-navigate-the-lunar-terrain-of-slope.

[10] Dec. 2022. URL: https://solarsystem.nasa.gov/resources/843/rare-full-moon-on-christmas-day?category=moons_earths-moon.

[11] Dec. 2022. URL: https://moon.nasa.gov/resources/476/lunar-swirl-reiner-gamma.

[12] T. Kanade, A. Yoshida, K. Oda, H. Kano, and M. Tanaka. "A stereo machine for video-rate dense depth mapping and its new applications." In: *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. June 1996, pp. 196–202. DOI: 10.1109/CVPR.1996.517074.

[13] Davide Scaramuzza and Friedrich Fraundorfer. "Visual Odometry [Tutorial]." en. In: *IEEE Robotics and Automation Magazine* 18.4 (Dec. 2011), pp. 80–92. ISSN: 1070-9932, 1558-223X. DOI: 10.1109/MRA.2011.943233.

[14] Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. "An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics." en. In: *Intelligent Industrial Systems* 1.4 (Dec. 2015), pp. 289–311. ISSN: 2363-6912, 2199-854X. DOI: 10.1007/s40903-015-0032-7.

[15] Davide Scaramuzza and Zichao Zhang. "Visual-Inertial Odometry of Aerial Robots." en. In: (), p. 13.

[16] *Springer Handbook of Robotics*. en. Springer Handbooks. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-32550-7. DOI: 10.1007/978-3-319-32552-1. URL: https://link.springer.com/10.1007/978-3-319-32552-1.

[17] en-US. URL: https://www.vicon.com/.

[18] en. URL: http://optitrack.com/index.html.

[19] D. Scharstein, R. Szeliski, and R. Zabih. "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms." en. In: *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*. Kauai, HI, USA: IEEE Comput. Soc, 2001, pp. 131–140. ISBN: 978-0-7695-1327-0. DOI: 10.1109/SMBV.2001.988771. URL: http://ieeexplore.ieee.org/document/988771/.

[20] Paul Furgale, Joern Rehder, and Roland Siegwart. "Unified temporal and spatial calibration for multi-sensor systems." In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Nov. 2013, pp. 1280–1286. DOI: 10.1109/IROS.2013.6696514.

[21] J. Kannala and S.S. Brandt. "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.8 (Aug. 2006), pp. 1335–1340. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2006.153.

[22] Christopher Mei and Patrick Rives. "Single View Point Omnidirectional Camera Calibration from Planar Grids." In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. Apr. 2007, pp. 3945–3950. DOI: 10.1109/ROBOT.2007.364084.

[23] Vladyslav Usenko, Nikolaus Demmel, and Daniel Cremers. "The Double Sphere Camera Model." In: *2018 International Conference on 3D Vision (3DV)*. Sept. 2018, pp. 552–560. DOI: 10.1109/3DV.2018.00069.

[24] Bogdan Khomutenko, Gaëtan Garcia, and Philippe Martinet. "An Enhanced Unified Camera Model." In: *IEEE Robotics and Automation Letters* 1.1 (Jan. 2016), pp. 137–144. ISSN: 2377-3766. DOI: 10.1109/LRA.2015.2502921.

[25] Donald B. Gennery. "Least-Squares Camera Calibration Including Lens Distortion and Automatic Editing of Calibration Points." In: *Calibration and Orientation of Cameras in Computer Vision*. Ed. by Armin Gruen and Thomas S. Huang. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 123–136. ISBN: 978-3-662-04567-1. DOI: 10.1007/978-3-662-04567-1_5. URL: https://doi.org/10.1007/978-3-662-04567-1_5.

[26] Kaichang Di and Rongxing Li. "CAHVOR camera model and its photogrammetric conversion for planetary applications." en. In: *Journal of Geophysical Research: Planets* 109.E4 (2004). ISSN: 2156-2202. DOI: 10.1029/2003JE002199. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2003JE002199.

[27] Gerda Kamberova and Ruzena Bajcsy. "Sensor Errors and the Uncertainties in Stereo Reconstruction." en. In: (), p. 21.

[28] Avishek Chatterjee and Venu Govindu. "Noise in Structured-Light Stereo Depth Cameras: Modeling and its Applications." In: (May 2015).

[29] Teledyne Flir. *Stereo Accuracy and Error Modeling*. URL: https://www.flir.com/globalassets/support/iis/application-notes/tan2004006_stereo_accuracy_error_modeling.pdf.

[30] Péter Fankhauser, Michael Bloesch, Christian Gehring, Marco Hutter, and Roland Siegwart. "Robot-Centric Elevation Mapping with Uncertainty Estimates." In: Sept. 2014. DOI: 10.1142/9789814623353_0051.

[31] Peter R. Florence, John Carter, Jake Ware, and Russ Tedrake. "NanoMap: Fast, Uncertainty-Aware Proximity Queries with Lazy Search Over Local 3D Data." en. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD: IEEE, May 2018, pp. 7631–7638. ISBN: 978-1-5386-3081-5. DOI: 10.1109/ICRA.2018.8463195. URL: https://ieeexplore.ieee.org/document/8463195/.

[32] Junjie Zeng, Rusheng Ju, Long Qin, Yue Hu, Quanjun Yin, and Cong Hu. "Navigation in Unknown Dynamic Environments Based on Deep Reinforcement Learning." In: *Sensors* 19 (Sept. 2019), p. 3837. DOI: 10.3390/s19183837.

[33] Panagiotis Papadakis. "Terrain traversability analysis methods for unmanned ground vehicles: A survey." en. In: *Engineering Applications of Artificial Intelligence* 26.4 (Apr. 2013), pp. 1373–1385. ISSN: 09521976. DOI: 10.1016/j.engappai.2013.01.006.

[34] Thomas Schöps, Torsten Sattler, and Marc Pollefeys. "SurfelMeshing: Online Surfel-Based Mesh Reconstruction." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.10 (Oct. 2020), pp. 2494–2507. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2019.2947048.

[35] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. "OctoMap: an efficient probabilistic 3D mapping framework based on octrees." en. In: *Autonomous Robots* 34.3 (Apr. 2013), pp. 189–206. ISSN: 0929-5593, 1573-7527. DOI: 10.1007/s10514-012-9321-0.

[36] Helen Oleynikova, Zachary Taylor, Marius Fehr, Juan Nieto, and Roland Siegwart. "Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-Board MAV Planning." In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Sept. 2017). arXiv: 1611.03631, pp. 1366–1373. DOI: 10.1109/IROS.2017.8202315.

[37] A. Yahja, A. Stentz, S. Singh, and B.L. Brumitt. "Framed-quadtree path planning for mobile robots operating in sparse environments." en. In: *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*. Vol. 1. Leuven, Belgium: IEEE, 1998, pp. 650–655. ISBN: 978-0-7803-4300-9. DOI: 10.1109/ROBOT.1998.677046. URL: http://ieeexplore.ieee.org/document/677046/.

[38] Renato Pajarola. "Overview of quadtree-based terrain triangulation and visualization." en. In: (2002). URL: https://escholarship.org/uc/item/24h2g848.

[39] Gerhard K. Kraetzschmar, Guillem Pagès Gassull, and Klaus Uhl. "Probabilistic quadtrees for variable-resolution mapping of large environments." en. In: *IFAC Proceedings Volumes* 37.8 (July 2004), pp. 675–680. ISSN: 14746670. DOI: 10.1016/S1474-6670(17)32056-6.

[40] A. Elfes. "Sonar-based real-world mapping and navigation." In: *IEEE Journal on Robotics and Automation* 3.3 (June 1987), pp. 249–265. ISSN: 2374-8710. DOI: 10.1109/JRA.1987.1087096.

[41] Alberto Elves. "Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation." In: *Tese de doutorado, Electrical and Computer Engineering, Carnegie Mellon University* (1989). URL: https://www.cs.cmu.edu/~motionplanning/papers/sbp_papers/integrated4/elfes_occup_grids.pdf.

[42] Johann Borenstein and Yoram Koren. "Real-time map building for fast mobile robot obstacle avoidance." en. In: ed. by Wendell H. Chun and William J. Wolfe. Boston, MA, 1991, pp. 74–81. DOI: 10.1117/12.25455. URL: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=954336.

[43] J. Borenstein and Y. Koren. "The vector field histogram-fast obstacle avoidance for mobile robots." In: *IEEE Transactions on Robotics and Automation* 7.3 (1991), pp. 278–288. ISSN: 2374-958X. DOI: 10.1109/70.88137.

[44] Wolfram Burgard, Armin B. Cremers, Dieter Fox, Dirk Hähnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner, and Sebastian Thrun. "Experiences with an interactive museum tour-guide robot." In: *Artificial Intelligence* 114.1 (1999), pp. 3–55. ISSN: 0004-3702. DOI: https://doi.org/10.1016/S0004-3702(99)00070-3.

[45] J. Borenstein and Y. Koren. "Histogramic in-motion mapping for mobile robot obstacle avoidance." In: *IEEE Transactions on Robotics and Automation* 7.4 (Aug. 1991), pp. 535–539. ISSN: 2374-958X. DOI: 10.1109/70.86083.

[46] Ulrich Raschke and Johann Borenstein. "A comparison of grid-type map-building techniques by index of performance." In: *Proceedings., IEEE International Conference on Robotics and Automation*. IEEE. 1990, pp. 1828–1832.

[47] Daniek Joubert, Willie Brink, and Ben Herbst. "Pose uncertainty in occupancy grids through Monte Carlo integration." In: *2013 16th International Conference on Advanced Robotics (ICAR)*. Nov. 2013, pp. 1–6. DOI: 10.1109/ICAR.2013.6766589.

[48] en. DOI: 10.1177/0278364916684382. URL: https://journals.sagepub.com/doi/epub/10.1177/0278364916684382.

[49] Mohd Nadhir Ab Wahab, Samia Nefti-Meziani, and Adham Atyabi. "A comparative review on mobile robot path planning: Classical or meta-heuristic methods?" en. In: *Annual Reviews in Control* 50 (2020), pp. 233–252. ISSN: 13675788. DOI: 10.1016/j.arcontrol.2020.10.001.

[50] Tixiao Shan, Jinkun Wang, Brendan Englot, and Kevin Doherty. "Bayesian Generalized Kernel Inference for Terrain Traversability Mapping." en. In: (), p. 10.

[51] Yiyuan Pan, Xuecheng Xu, Yue Wang, Xiaqing Ding, and Rong Xiong. "GPU accelerated real-time traversability mapping." In: *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. Dec. 2019, pp. 734–740. DOI: 10.1109/ROBIO49542.2019.8961816.

[52] H. Seraji. "Traversability index: a new concept for planetary rovers." In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*. Vol. 3. May 1999, 2006–2013 vol.3. DOI: 10.1109/ROBOT.1999.770402.

[53] Hans P Moravec. "Robot Spatial Perception by Stereoscopic Vision and 3D Evidence Grids." en. In: (), p. 44.

[54] M. Daily, J. Harris, D. Keirsey, D. Olin, D. Payton, K. Reiser, J. Rosenblatt, D. Tseng, and V. Wong. "Autonomous cross-country navigation with the ALV." In: *1988 IEEE International Conference on Robotics and Automation Proceedings*. Apr. 1988, 718–726 vol.2. DOI: 10.1109/ROBOT.1988.12144.

[55] I.S. Kweon and T. Kanade. "High-resolution terrain map from multiple sensor data." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (Feb. 1992), pp. 278–292. ISSN: 1939-3539. DOI: 10.1109/34.121795.

[56] Regis Hoffman and Eric Krotkov. "Terrain Roughness Measurement from Elevation Maps." en. In: Philadelphia, PA, United States, Mar. 1990, p. 104. DOI: 10.1117/12.969874. URL: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.969874.

[57] Pedro F Proença, Jeff Delaune, and Roland Brockers. "Optimizing Terrain Mapping and Landing Site Detection for Autonomous UAVs." en. In: (), p. 7.

[58] Cang Ye. "Navigating a Mobile Robot by a Traversability Field Histogram." In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 37.2 (Apr. 2007), pp. 361–372. ISSN: 1941-0492. DOI: 10.1109/TSMCB.2006.883870.

[59] Juil Sock, Jun Kim, Jihong Min, and Kiho Kwak. "Probabilistic traversability map generation using 3D-LIDAR and camera." In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. May 2016, pp. 5631–5637. DOI: 10.1109/ICRA.2016.7487782.

[60] Péter Fankhauser, Michael Bloesch, and Marco Hutter. "Probabilistic Terrain Mapping for Mobile Robots With Uncertain Localization." In: *IEEE Robotics and Automation Letters* 3.4 (Oct. 2018), pp. 3019–3026. ISSN: 2377-3766. DOI: 10.1109/LRA.2018.2849506.

[61] Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. "Learning robust perceptive locomotion for quadrupedal robots in the wild." en. In: *Science Robotics* 7.62 (Jan. 2022), eabk2822. ISSN: 2470-9476. DOI: 10.1126/scirobotics.abk2822.

[62] Rudolph Triebel, Patrick Pfaff, and Wolfram Burgard. "Multi-Level Surface Maps for Outdoor Terrain Mapping and Loop Closing." en. In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Beijing, China: IEEE, Oct. 2006, pp. 2276–2282. ISBN: 978-1-4244-0258-8. DOI: 10.1109/IROS.2006.282632. URL: http://ieeexplore.ieee.org/document/4058725/.

[63] Jacek Zienkiewicz, Akis Tsiotsios, Andrew Davison, and Stefan Leutenegger. "Monocular, Real-Time Surface Reconstruction Using Dynamic Level of Detail." In: *2016 Fourth International Conference on 3D Vision (3DV)*. Oct. 2016, pp. 37–46. DOI: 10.1109/3DV.2016.82.

[64] Clint D. Lombard and Corné E. van Daalen. "Stochastic triangular mesh mapping: A terrain mapping technique for autonomous mobile robots." In: *Robotics and Autonomous Systems* 127 (May 2020). arXiv: 1910.03644, p. 103449. ISSN: 09218890. DOI: 10.1016/j.robot.2020.103449.

[65] Donald B Gennery. "Traversability Analysis and Path Planning for a Planetary Rover." en. In: (), p. 16.

[66] S. Singh, R. Simmons, T. Smith, A. Stentz, V. Verma, A. Yahja, and K. Schwehr. "Recent progress in local and global traversability for planetary rovers." In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*. Vol. 2. Apr. 2000, 1194–1200 vol.2. DOI: 10.1109/ROBOT.2000.844761.

[67] D. Langer, J.K. Rosenblatt, and M. Hebert. "A behavior-based system for off-road navigation." In: *IEEE Transactions on Robotics and Automation* 10.6 (Dec. 1994), pp. 776–783. ISSN: 2374-958X. DOI: 10.1109/70.338532.

[68] J.W. Weingarten, G. Gruener, and R. Siegwart. "Probabilistic plane fitting in 3D and an application to robotic mapping." In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*. Vol. 1. Apr. 2004, 927–932 Vol.1. DOI: 10.1109/ROBOT.2004.1307268.

[69] Chanoh Park, Soohwan Kim, Peyman Moghadam, Clinton Fookes, and Sridha Sridharan. "Probabilistic Surfel Fusion for Dense LiDAR Mapping." en. In: arXiv:1709.01265 (Sept. 2017). arXiv:1709.01265 [cs]. URL: http://arxiv.org/abs/1709.01265.

[70] Hanspeter Pfister, Matthias Zwicker, Jeroen van Baar, and Markus Gross. "Surfels: surface elements as rendering primitives." en. In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*. Not Known: ACM Press, 2000, pp. 335–342. ISBN: 978-1-58113-208-3. DOI: 10.1145/344779.344936. URL: http://portal.acm.org/citation.cfm?doid=344779.344936.

[71] Thomas Whelan, Stefan Leutenegger, Renato Salas Moreno, Ben Glocker, and Andrew Davison. "ElasticFusion: Dense SLAM Without A Pose Graph." en. In: *Robotics: Science and Systems XI*. Robotics: Science and Systems Foundation, July 2015. ISBN: 978-0-9923747-1-6. DOI: 10.15607/RSS.2015.XI.001. URL: http://www.roboticsproceedings.org/rss11/p01.pdf.

[72] Jörg Stückler and Sven Behnke. "Multi-resolution surfel maps for efficient dense 3D modeling and tracking." en. In: *Journal of Visual Communication and Image Representation* 25.1 (Jan. 2014), pp. 137–147. ISSN: 10473203. DOI: 10.1016/j.jvcir.2013.02.008.

[73] Luxin Han, Fei Gao, Boyu Zhou, and Shaojie Shen. "FIESTA: Fast Incremental Euclidean Distance Fields for Online Motion Planning of Aerial Robots." In: *arXiv:1903.02144 [cs]* (July 2019). arXiv: 1903.02144. URL: http://arxiv.org/abs/1903.02144.

[74] Richard A Newcombe, Andrew J Davison, Shahram Izadi, Pushmeet Kohli, Otmar Hilliges, Jamie Shotton, David Molyneaux, Steve Hodges, David Kim, and Andrew Fitzgibbon. "KinectFusion: Real-Time Dense Surface Mapping and Tracking." en. In: (), p. 10.

[75] Victor Reijgwart, Alexander Millane, Helen Oleynikova, Roland Siegwart, Cesar Cadena, and Juan Nieto. "Voxgraph: Globally Consistent, Volumetric Mapping Using Signed Distance Function Submaps." In: *IEEE Robotics and Automation Letters* 5.1 (Jan. 2020), pp. 227–234. ISSN: 2377-3766. DOI: 10.1109/LRA.2019.2953859.

[76] Lukas Schmid, Victor Reijgwart, Lionel Ott, Juan Nieto, Roland Siegwart, and Cesar Cadena. "A Unified Approach for Autonomous Volumetric Exploration of Large Scale Environments Under Severe Odometry Drift." In: *IEEE Robotics and Automation Letters* 6.3 (July 2021), pp. 4504–4511. ISSN: 2377-3766. DOI: 10.1109/LRA.2021.3068954.

[77] Xian Guo and Yongchun Fang. "Learning to Navigate in Unknown Environments Based on GMRP-N." In: *2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*. July 2019, pp. 1453–1458. DOI: 10.1109/CYBER46603.2019.9066637.

[78] Jonas Frey, Hermann Blum, Francesco Milano, Roland Siegwart, and Cesar Cadena. "Continual Adaptation of Semantic Segmentation Using Complementary 2D-3D Data Representations." In: *IEEE Robotics and Automation Letters* 7.4 (Oct. 2022), pp. 11665–11672. ISSN: 2377-3766. DOI: 10.1109/LRA.2022.3203812.

[79] Brandon Rothrock, Ryan Kennedy, Chris Cunningham, Jeremie Papon, Matthew Heverly, and Masahiro Ono. "SPOC: Deep Learning-based Terrain Classification for Mars Rover Missions." In: *AIAA SPACE 2016*. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2016-5539. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2016-5539.

[80] Lorenz Wellhausen and Marco Hutter. "Rough Terrain Navigation for Legged Robots using Reachability Planning and Template Learning." In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sept. 2021, pp. 6914–6921. DOI: 10.1109/IROS51168.2021.9636358.

[81] Pascal Schoppmann, Pedro F. Proença, Jeff Delaune, Michael Pantic, Timo Hinzmann, Larry Matthies, Roland Siegwart, and Roland Brockers. "Multi-Resolution Elevation Mapping and Safe Landing Site Detection with Applications to Planetary Rotorcraft." In: *arXiv:2111.06271 [cs]* (Nov. 2021). arXiv: 2111.06271. URL: http://arxiv.org/abs/2111.06271.

[82] Tony Finch. "Incremental calculation of weighted mean and variance." en. In: (), p. 8.

[83] Patrick Geneva Guoquan Huang. *vicon2gt: Derivations and Analysis*. en-US. Dec. 2021. URL: https://raw.githubusercontent.com/rpng/vicon2gt/master/docs/tr_vicon2gt.pdf.

[84] Joern Rehder, Janosch Nikolic, Thomas Schneider, Timo Hinzmann, and Roland Siegwart. "Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes." In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. May 2016, pp. 4304–4311. DOI: 10.1109/ICRA.2016.7487628.

[85] Wayne Johnson et al. "Mars Science Helicopter Conceptual Design." en. In: (2020).

Die Schriftenreihe

wird vom Lehrstuhl für Informatik XVII: Robotik der Universität Würzburg herausgegeben und präsentiert innovative Forschung aus den Bereichen der Robotik und der Telematik.

Die Kombination fortgeschrittener Informationsverarbeitungsmethoden mit Verfahren der Regelungstechnik eröffnet hier interessante Forschungs- und Anwendungsperspektiven. Es werden dabei folgende interdisziplinäre Aufgabenschwerpunkte bearbeitet:

- Sensorik: Integration von Sensoren in robotische Systeme, Kalibrierung, Lokalisierung, Kartierung und Interpretation von Sensordaten in Echtzeit.

- Robotik und Mechatronik: Kombination von Informatik, Elektronik, Mechanik, Aktuatorik, Regelungs- und Steuerungstechnik, um Roboter adaptiv und flexibel ihrer Arbeitsumgebung anzupassen.

Anwendungsschwerpunkte sind u.a. mobile Roboter, Tele-Robotik, Raumfahrtsysteme und Medizin-Robotik.

Lehrstuhl Informatik XVII
Robotik
Am Hubland
D-97074 Wuerzburg

Tel.: +49 (0) 931 - 31 - 88790

andreas.nuechter@uni-wuerzburg.de
https://www.informatik.uni-wuerzburg.de/robotics

ISSN: 2940-6145 (online)
ISSN: 2940-6137 (print)
ISBN: 978-3-945459-49-2 (online)

Zitation dieser Publikation

WERNER, L. (2024). Terrain Mapping for Autonomous Navigation of Lunar Rovers. Schriftenreihe Forschungsberichte in der Robotik, Band 29. Würzburg: Universität Würzburg.
DOI: 10.25972/OPUS-35826