How Rough is the Path? Terrain Traversability Estimation for Local and Global Path Planning

Gabriel G. Waibel[1,2], Tobias Löw[1,2], Mathieu Nass[1,3], David Howard[1], Tirthankar Bandyopadhyay[1], Paulo V. K. Borges[1]

***Abstract*—Perception and interpretation of the terrain is essential for robot navigation, particularly in off-road areas, where terrain characteristics can be highly variable. When planning a path, features such as the terrain gradient and roughness should be considered, and they can jointly represent the traversability cost of the terrain. Despite this range of contributing factors, most cost maps are currently binary in nature, solely indicating traversible versus non-traversible areas. This work presents a joint local and global planning methodology for building continuous cost maps using LIDAR, based on a novel traversability representation of the environment. We investigate two approaches. The first, a statistical approach, computes terrain cost directly from the point cloud. The second, a learning-based approach, predicts an IMU response solely from geometric point cloud data using a 2D-Convolutional-LSTM neural network. This allows us to estimate the cost of a patch without directly driving over it, based on a data set that maps IMU signals to point cloud patches. Based on the terrain analysis, two continuous cost maps are generated to jointly select the optimal path considering distance and traversability cost for local navigation. We present a real-time terrain analysis strategy applicable for local planning, and furthermore demonstrate the straightforward application of the same approach in batch mode for global planning. Off-road autonomous driving experiments in a large and hybrid site illustrate the applicability of the method. We have made the code available online for users to test the method.**

I. INTRODUCTION

Autonomous navigation is a key research area which underpins ongoing advances in autonomous driving. A critical feature for autonomous navigation is generation of a cost map which is used by a path planner to computes the optimal path from the current position to the desired destination. Cost maps are traditionally considered in the literature as a discrete representation, where the environment is divided into traversable and non-traversable areas. Although this approach performs well in both indoor [1] and on-road [2], [3] settings, off-road terrain is often too heterogeneous to usefully distinguish between these two classes. Additionally, an obstacle in an urban environment, such as a curbstone, may not be an obstacle on natural grounds, where the robot must traverse over natural undulations of similar size and shape to reach the goal. Therefore, the terrain should be represented with a continuous traversability score, which influences the planned path alongside other metrics (e.g., distance to the goal).

[1]Robotics and Autonomous System Group, Data61, CSIRO, Australia david.howard,paulo.borges,tirtha.bandy@csiro.au

[2]Autonomous Systems Lab, ETH Zurich, 8092 Zurich, Switzerland waibelg@student.ethz.ch

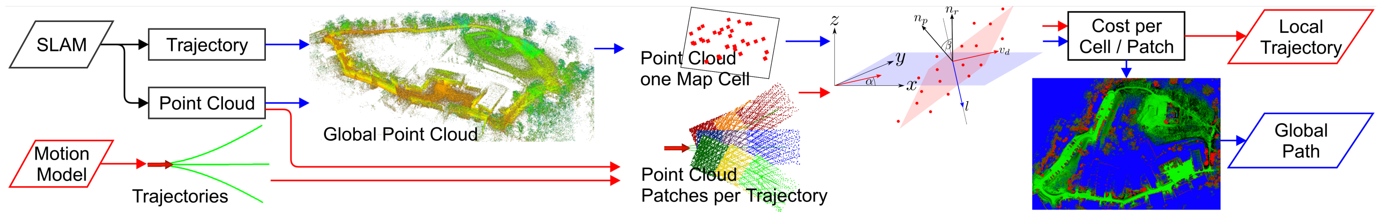
[3]Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, Netherlands mathieunass1@gmail.com

The main idea of terrain analysis is based on the fact that, in general, steep and rough ground should be avoided and therefore lead to higher costs than flat and level surfaces. Challenging terrain should be avoided whenever possible since it can increase the chance of mission failure, it can cause material stress, lead to dicomfort of any passengers, and may require more energy. For this reason, a longer but smoother path is preferable in certain situations. Steep terrain is relatively easy to perceive and compute [4], but rough terrain remains challenging to interpret, especially if the ground is covered in high grass or driving occurs in low-visibility environments.

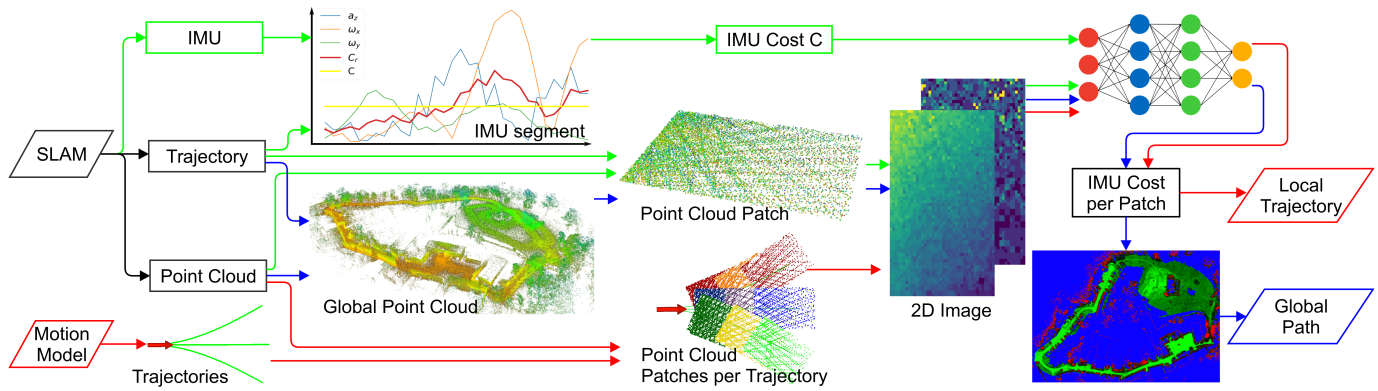
Terrain analysis methods that estimate the ground from image data have achieved good results in the past [5], [6], [7], [8]; however more recent approaches apply machine learning methods to gain a more comprehensive understanding of the scene. Supervised learning approaches segment the environment into different classes using human-provided labels [9], [10], [11], [12]. Since terrain features such as inclination, roughness or slippage can be estimated by various sensors, the robot is able to learn them from its own experience [13], [14], [15], [16]. Many state-of-the-art learning approaches based on neural networks (NN) use image data since powerful Convolutional Neural Networks (CNNs) are directly applicable to images. There are situations, however, in which varying/poor illumination conditions can deteriorate the results, making range sensors such as LiDAR a strong alternative.

This work focuses on navigation cost map generation for an Autonomous Ground Vehicle (AGV) in mixed (paved and off-road) environments. Cost maps are generated through two main approaches; a Statistical-based Roughness Estimation (SRE) and a Learning-based Roughness Estimation (LRE) method. An overview of each is shown in Figures 1(a) and 1(b). SRE uses LiDAR data to estimate the terrain inclination and roughness directly from the point cloud. For LRE, we develop an end-to-end solution, from data extraction to cost assignment. The underlying concept is that terrain roughness is experienced by the vehicle when interacting with the surface, which can be efficiently measured by proprioceptive sensing with an Inertial Measurement Unit (IMU). By matching IMU segments to corresponding point cloud regions, we are able to learn a mapping from point cloud to IMU signal, and subsequently estimate costs of unseen regions of the map via exteroceptive LiDAR readings only. Point cloud regions are treated as images and are processed directly by a temporally-sensitive CNN.

Both approaches are applied to local and global planning. For local planning, multiple trajectories are generated, and



(a) Statistical Roughness Estimation (SRE): The blue arrows illustrate the data flow for global planning and the red ones for local planning. SLAM is used to register incoming point cloud messages in a global point cloud. The point cloud is discretized into a 2D-grid receiving a point cloud for each cell. The point cloud is analyzed for its statistical properties resulting in a cost per cell on the global map. For local planning, multiple trajectories are generated based on the vehicle motion model. On each trajectory, multiple point cloud patches are extracted and analyzed based on their variances in the vertical direction and their gradients. The trajectory with the lowest cost is selected.



(b) Learning-based Roughness Estimation (LRE): The green arrows are for NN training, red for local planning, and blue for global planning. For NN training, the C-SLAM algorithm is running on the data and provides the sensor trajectory, IMU measurements and the point cloud of the environment. The poses of the trajectory are used to couple a point cloud patch with the IMU segment recorded around the same timestamp. The IMU cost is computed from an IMU segment and it is used to label the associated point cloud patch, which is down-casted to a multi-dimensional image. For the global map, the C-SLAM algorithm generates a global point cloud from which point cloud patches are extracted. The point cloud patch is down-casted to a multi-dimensional image and the NN predicts the IMU cost per image. The resulting IMU cost is used to generate a 2D cost map, which is required for global planning. For local planning, multiple trajectories are generated based on the vehicle motion model. Point cloud patches are constructed from the online point cloud and down-casted to images for each trajectory. All images of one trajectory are fed into the NN predicting the IMU cost per trajectory. The trajectory with the lowest cost is selected.

Fig. 1. Overview of data flow for the two proposed approaches (SRE/LRE): The global maps have obstacles represented in red and unknown cells in blue. The terrain traversability is encoded continuously from light green (easy to traverse) to dark green and black (hard to traverse, but still traversable, unlike obstacles in red).

each trajectory is ranked according to the distance to the goal and the terrain traversability using SRE or LRE in real-time. The best trajectory is selected based on a combination of distance and terrain cost. The applicability of the algorithms is illustrated through autonomous driving with a full-size off-road vehicle in field experiments, and numerous characteristics are calculated based on measured real-world data, providing us various metrics for the smoothness of the ground. Both methods are compared to each other and to a traditional Binary Method (BM), which divides the environment into traversable and non-traversable areas.

In addition, a continuous global cost map is generated from point cloud data for each approach. Instead of using a binary representation for planning, the proposed metric maps represent the terrain using 8-bits. In order to evaluate the maps, in our implementation we use the Dijkstra algorithm to select the optimal path based on terrain and distance, and the paths on both maps are compared against each other and against the paths on an occupancy grid, namely the binary method (BM) by the same path evaluation as for the local case.

Results indicate that SRE and LRE far exceed the performance of BM, and display high performance as general cost map generators for navigation that are readily applicable to both global and local planning. Field experiments show the generation of high quality paths across a number of evaluation scenarios when driving autonomously.

II. Related Work

In early works, robots were mostly reliant on a purely local planning approaches [5], [17], which can cause them to get constrained in local minima, such as dead ends. Global maps provide an alternative for determining a suitable path based on a previously contructed representation of the environment. Thrun [18] differentiates between metric [19] and topological maps [20], [21], where metric maps provide a linear and continuous representation of the world and topological maps indicate landmark points of interest in the environment. In our work, we focus on metric maps, since they are easier to construct and allow for optimal global paths to be determined.

Most of the literature in terrain estimation is divided into approaches analysing the terrain analytically and methods based on learning. As our work is based on LiDAR data, the following two subsections review some key methods for terrain analysis based on geometrical and range data.

A. Statistical Methods

In general, traversability is a function of the terrain geometry and the terramechanic (properties of the soil and their interaction with the vehicle wheels/tracks) characteristics [22], [23]. The terrain geometry can be simplified by slope and roughness, as discussed in early works that extract basic terrain statistics (variance and slope of patches in front of the vehicle) to quantify the traversability cost [5], [24], [25]. Roughness is the small-scale variations in the height of a physical surface, being closely related to traversability [26]. Mathematically, roughness depends on how scattered or linear/planar the distribution of the points in the area of interest is. Roughness can be quantified using a number of statistical methods such as least squares plane fitting computing the residuals [24], [27], Gaussian mixture models and principal component analysis over the terrain points [28]. The illustration in Figure 2 shows the concept for 2D LiDAR returns for a vehicle-sized patch. A 'smoother' terrain is shown on the top image, according to the residuals and slope of the LiDAR readings. Hence, the distribution of points in the top figure arguably correspond to a terrain that is easier to traverse than bottom image. For the slope analysis, the concept should obviously consider both pitch and roll, as both influence traversability (Figure 3).

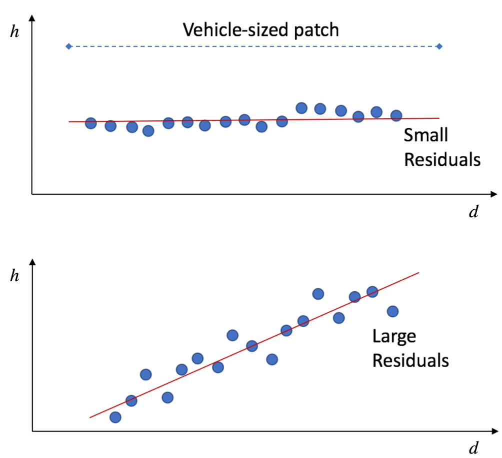


Fig. 2. Concept illustration of LiDAR readings (blue dots) for distinct terrains. d represents the distance from the sensor and h the height with respect to the sensor frame. The top image represents a 'smoother' terrain, with minimal roughness (small residuals for the red line fitting) on horizontal ground. The bottom image shows a rougher surface (larger residuals in the red line fitting) with more slope.

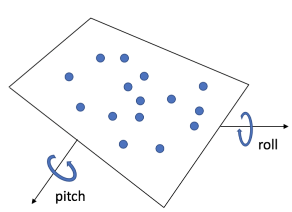


Fig. 3. Concept illustration of LiDAR readings (blue dots) a inclined surface. The traversability metric should include both pitch and roll.

Terrain roughness can be also estimated with the standard deviation of multiple plane-fits [8], the distribution of point cluster by calculating their covariance matrix [4], the roll and pitch angle combined with the residual in the vertical component z [6], the variance in z [29], or the height of the largest step within predefined area [27]. Some approaches use fuzzy logic to merge multiple properties [30], [31]. There is still potential for improvement by combining various properties and multiple resolutions. Although the terrain is estimated continuously, ultimately the values are classified into traversable and untraversable regions, and the path is planned on this binary representation. Various works have demonstrated the IMU's reading association to the type of ground under the vehicle. The inertial data is often used for classification of terrain [32]. The performance can be improved by considering the speed dependency of the IMU [33]. IMU sensors mounted on cars [34], [35](statistical) are used to estimate the road roughness or detecting bumps on roads. None of the approaches forecasts a continuous traversability score from a point cloud patch relying on the IMU.

B. Learning Methods

Learning-based methods generally aim to gain a semantic understanding in order to interpret the environment, but can also used for traversability cost regression. The most common application is segmentation or classification of the environment. Point cloud scenes are segmented semantically into different classes such as road, gravel, sand, grass, for example [36], [37], [38], or into traversable and non-traversable areas [39], [40], [41]. For both, the intra-class variation can be quite high, and no information is provided on how challenging the terrain is in a traversable class. In contrast, our proposed regression model predicts a continuous terrain traversability cost only from point cloud data. Recently, NNs achieved impressive results on general classification tasks [42], [43], [44]. As part of this work we evaluated PointNet [42] for the terrain analysis task, but the differences between natural terrain patches are much smaller than the differences between well defined objects (e.g., a table and a chair). In [45] terrain is classified by the means of inertial data on indoor grounds. Similarly to this paper, the work by Oliveira et al [46] performs terrain analysis for global planning using deep learning networks as a classification task. However, our method uses a temporally-sensitive network representation that allows for temporal patterns to be harnessed by the model, performing a continuous regression. Additionally, we apply the LRE in real-time for both local and global planning and compare it against the proposed SRE statistical approach.

In summary, our work advances the state of the art by predicting a continuous terrain cost solely from geometrical LiDAR data and the continuous values are used for global and local planning resulting in a smoother path.

III. BACKGROUND

In this section we provide information on background algorithms that are used for terrain analysis and local planning. In particular, we describe the LiDAR SLAM system employed, and trajectory generation method for local planning.

A. SLAM

For the terrain analysis, information from LiDAR is used to estimate traversability costs. The points are registered into a coherent point cloud prior to analysis using the CSIRO-SLAM (C-SLAM) algorithm as fully described in [47], [48]. C-SLAM performs 3D scan-matching. Until convergence, corresponding point cloud features are matched (correspondence step). The robot's trajectory is then optimized to minimize errors between the matched features and the deviation from the measured inertial data (optimization step).

The correspondence step matches surfels of consecutive LiDAR scans. Point clusters are created based on the point's temporal and spatial information. If there are enough points in one set, the eigenvalues of its second-moment matrix determine the cluster's distribution. The surfel's position and normals form a 6D vector. Surfel matches are obtained from a k-nearest neighbor search of a *kd*-tree, and the error between the matches is computed.

The optimization step solves a linear optimization problem to receive the trajectory corrections $\delta_r(\tau_i)$, $\delta_t(\tau_i)$ at the sampled times τ_i:

$$\begin{bmatrix} A_{match} \\ \hdashline A_{smooth} \\ \hdashline A_{initial} \end{bmatrix} \begin{bmatrix} \delta_r(\tau_1) \\ \delta_t(\tau_1) \\ \vdots \\ \delta_r(\tau_n) \\ \delta_t(\tau_n) \end{bmatrix} = \begin{bmatrix} b_{match} \\ \hdashline b_{smooth} \\ \hdashline b_{initial} \end{bmatrix} \tag{1}$$

A and b are linear constraints: surfels match constraints, discretized trajectory smoothness constraints, and initial conditions ensuring continuity with the previous trajectory. It is solved by the M-estimator framework using Cauchy weights, and the trajectory is reconstructed by a cubic spline. These steps are repeated until convergence between the matched surfels and the estimated trajectory is achieved.

SLAM is used during data extraction for the learning approach to couple point cloud patches with the appropriate IMU data. Besides, SLAM registers the recorded points of the entire operation site into a global point cloud, which serves as the starting point for the generation of the global cost maps by the statistical and the learning-based approach.

B. Local Trajectory Generation

For local planning, numerous possible trajectories are generated and rated based on terrain cost. Trajectory generation is based on the kinematic constraints of the non-holonomic vehicle, which is modelled as an Ackermann steering platform. The robot's state (position x,y, orientation θ) evolves according to the following forward kinematics model:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} v\ \cos(\theta(t)) \\ v\ \sin(\theta(t)) \\ \frac{v}{L}\ \tan(\varphi) \end{bmatrix} \tag{2}$$

L is the wheelbase of the robot. The inputs are the velocity v and the steering angle φ, which are constrained by the vehicle properties. The equation is executed in its discretized form with a $\Delta t = 0.1\,\mathrm{s}$. For illustration, three possible trajectories computed according to equation (2) are shown on the left image of Figure 7. For our local planning algorithm, multiple trajectories are generated at discretized steering angles and are rated based on the sensed terrain at each planning step, as discussed in the following sections.

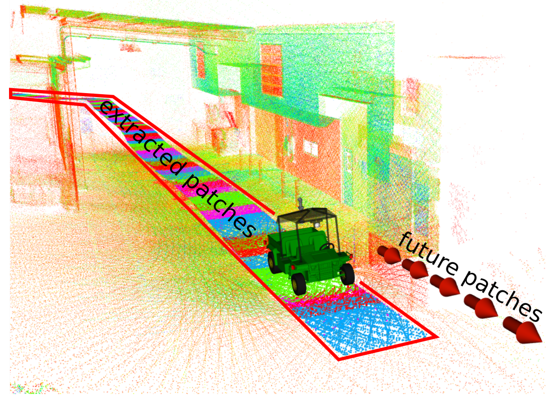


Fig. 4. Patch frames are allocated in front of the vehicle and follow its planned trajectory. Points from LIDAR readings that fall within a frame are allocated to that frame, otherwise they are discarded. Subsequent sweeps as the vehicle approaches a patch may add new points to the patch and change the value of terrain characteristics. The latest values of these patches are used for decision making. Once the vehicle drives over the patch, the IMU sensation is matched to the patch and saved, and the patch frame is reallocated to the end of the frame queue. Points can be then allocated to it on the next LIDAR sweep. The vehicle follows the trajectory denoted by red arrows, previously-extracted patches are visualised behind the vehicle.

IV. Methodology

In this section we describe the entire learning pipeline from data extraction to the SRE and to the NN architecture for LRE. We also explain how our terrain analysis methods improve the quality of the local trajectory selection (local path planning). Finally, we use both methods to generate two continuous cost maps for global planning.

A. Prediction of the IMU Cost with a Convolutional-LSTM Network

Our LRE approach predicts IMU costs from a point cloud patch. The NN is trained in a self-supervised fashion based on data recorded through manual driving.

1) Data Extraction: The method extracts point cloud patches slightly larger than the vehicle's footprint (for our platform shown in Figure 8, the patch is $3\,\mathrm{m} \times 2\,\mathrm{m}$) and pairs them with corresponding IMU data acquired when the vehicle traverses that patch, as shown in Figure 4. During data recording, the C-SLAM algorithm aligns the coordinate frames for the three types of information: trajectory T, point cloud P, and inertial data I. The vehicle trajectory is represented by a vector of poses and it is used to couple an IMU segment I_k of length N with a point cloud patch P_k at location $T[k]$. As inertial data is speed-dependent [33], each patch has an associated speed measurement S_k.

a) Down-Casting: The point cloud patch is flattened to a multi-channel image before applying convolutional filters which are part of the NN processing. Each point in the patch

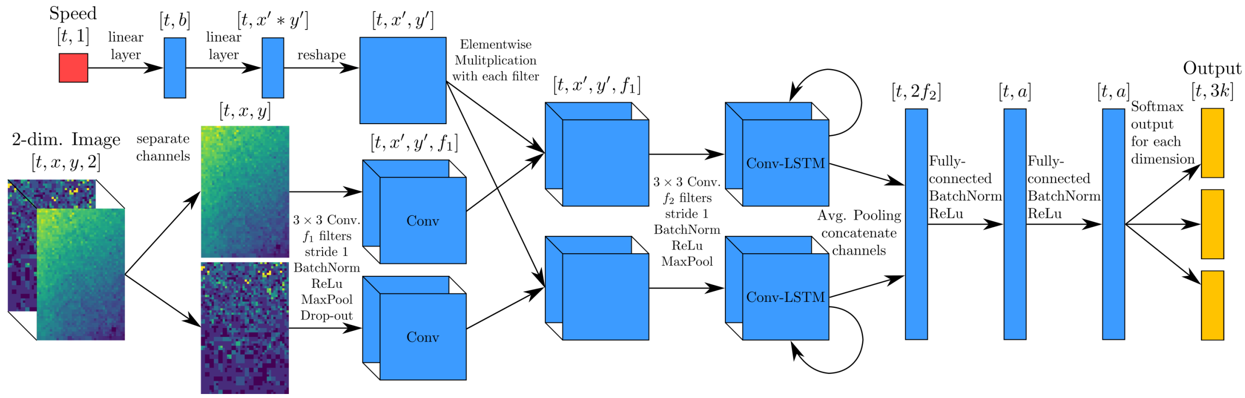


Fig. 5. Deep Convolutional-LSTM Network Architecture: a simplified representation showing a 2-dimensional image and a 1-dimensional output. The image has dimension $(x,y,2)$, where x, and y are the height and width of the image, and 2 is the input's dimensionality (mean and s.d.in z). Practically, we feed in a sequence of t patches and speeds to the LSTM, and predict the corresponding t IMU measurement segments. Input channels are separated and processed by convolutional modules; each module consists of a convolution layer (number of filters f_1, kernel size 3, stride 1), a batch normalization layer, a ReLu activation function, a max-pooling layer (kernel size 2) and a drop-out layer (rate: 0.3). A fully-connected network transforms the t speed inputs to the dimension of the image after the convolution filter. The linear layer is reshaped and forms an activation map of the side-input, which is multiplied element-wise with each filter of the resulting feature map of the image channels. The resulting images are fed into multiple 2D-Convolutional-LSTM modules, the last of which contains f_2 filters. Average pooling is applied to each channel and the channels are concatenated. After processing by a fully-connected layer of size a, a batch normalization layer, and a ReLu activation function, the last hidden layer is fully-connected to each output dimension through a softmax function.

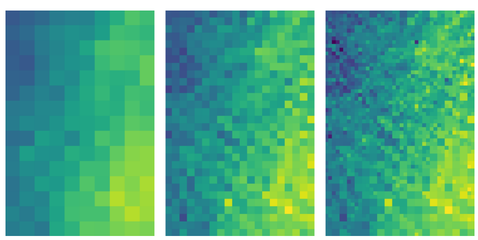


Fig. 6. Possible figure pyramidal approach: The three different resolutions for the mean channel.

is assigned to its closest pixel in the image. The brightness values in the image correspond to the height values in the 3D points. The mean z-value μ_z, the standard deviation in the z direction σ_z, and the difference between the maximum and minimum z-value for all points in each pixel are calculated, giving a 3-dimensional image.

Since not all pixels may have points assigned, a pyramidal approach is implemented. Three images of different resolutions (0.2 m, 0.1 m, and 0.05 m) are queried; if a pixel in the highest resolution image is empty, the value of the next lowest resolution image is used instead. The pixel of the lowest resolution image is never empty because blank pixels are interpolated. In practice, the lowest resolution image is usually not required as the images are generated from $\approx$ 3500-4500 points, which are relatively uniformly distributed.

b) IMU Transformation: Each image is a discrete time snapshot of the terrain, whereas its associated IMU segment is a continuous-time signal. The objective of the IMU transformation is to label an image with one continuous terrain traversability score, which can be used for planning with no post-processing required.

We propose a combined IMU cost C, consisting of the angular velocity in x and y and linear acceleration in z, represented by ω_x, ω_y and a_z, respectively. Linear accelerations in x and y are neglected as they vary due to vehicle behaviour (e.g., accelerating or turning sharply), and are not necessarily dependent on terrain. This is also true for angular velocity in z.

A constant IMU bias is subtracted. At first, the absolute value is taken for each segment (ω_x, ω_y, or a_z) because the cost is not dependent on polarity, and each segment is normalized into the range $[0,1]$. All three segments are averaged, and the real IMU cost C_r of length N is obtained:

$$C_r[N] = \frac{\omega_x[N] + \omega_y[N] + a_z[N]}{3} \tag{3}$$

C_r is a smoother, more stable signal than a single IMU segment. Since the IMU cost per patch is desired, the mean value over all N coefficients of C_r is calculated:

$$C = \frac{1}{N}\sum_{N} C_r[N] \tag{4}$$

C is directly associated to the terrain traversability cost and is used for navigation.

2) Deep 2D-Convolutional-LSTM Network: In order to regress the traversability cost from IMU readings, we use a Conv-LSTM network as illustrated in Figure 5. Convolutional layers are used because the point cloud patches are transformed to images. LSTM modules are beneficial on entire sequences of data. Since the point cloud patches and the IMU segments are extracted from the driven trajectory, neighboring samples are similar to each other. Therefore feeding a full sequence into the NN increases its performance. Speed is provided as a side input to the network, as the IMU signal (especially in rough terrain) is highly speed-dependent [33]. Speed is incorporated via an activation map [49], where in fully-connected layers learn a transformation from the dimension of the side-input to the dimension of the feature map. The side-channel is multiplied element-wise with each filter of the feature map after convolution. Softmax is applied at each output node, providing a sparse representation of the data to facilitate training [50]. k=10 softmax values represent one output dimension.

B. Local Terrain Analysis

Algorithm 1: Local planning

Input: Current robot state r, global goal g, vehicle's motion model m

Output: Lowest cost trajectory.

while not at g **do**
  get local goal $l(r,g)$;
  generate T trajectories(r, l, m) (figure 7);
  foreach trajectory t **do**
    init p patches $\leftarrow$ 3 poses on t;
    foreach p **do**
      foreach point in local point cloud **do**
        if point in p's boundaries **then**
          add point to p (figure 7);
    statistical $\leftarrow$ 0;
    foreach p **do**
      statistical $+=(\nu+\beta)$;
      down-cast p's point cloud to image;
    learning $\leftarrow$ NN(image sequence);
    cost $\leftarrow$ distance(t,l) + (statistical | learning);
  trajectory $\leftarrow \arg\min$(cost);

The local planner adapts the global path by rating multiple trajectories based on the locally sensed terrain characteristics, deviating around large irregularities as required (Algorithm 1).

At each planning step, the algorithm generates T trajectories based on the vehicle motion model as described in Section III-B. Three patch frames are allocated per trajectory and filled from the local point cloud. Terrain cost is per trajectory is computed by one of the two methods:

1) SRE: Compute ν in z and β of each patch in real-time using the RANSAC algorithm. Terrain cost is the summed mean of both across all patches in the trajectory.
2) LRE: Down-cast each patch into an image, and feed the trajectory's image sequence into the NN. The network predicts C per trajectory, which has the same length as the image sequence. Terrain cost is the mean value of the output sequence.

All costs are normalized. Since the area with the most accessible terrain should be selected, a polynomial of degree d=4 is fitted through each terrain cost. The final cost per trajectory is a user-defined weighted average between the terrain cost and the Euclidean distance cost.

C. Map Generation

This section describes the generation of the global cost maps, using both SRE and LRE. The input to both algorithms is a global point cloud generated by C-SLAM, which is subdivided into 20×20cm cells. Costs per cell range from 0 (free-space) to 255 (obstacle).

1) Statistical Roughness Estimation SRE: Each point of the global point cloud is assigned to a cell. Cells characteristics are then calculated on all points per cell as follows.

First, all points in one of the 2D cell are formed into a matrix G, where

$$G=\begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_n \\ y_1 & y_2 & y_3 & \cdots & y_n \\ z_1 & z_2 & z_3 & \cdots & z_n \end{bmatrix} \tag{5}$$

The roughness is then estimated by calculating the variance of all points in z-direction.

$$\nu=\sigma(G[3,:]) \tag{6}$$

and the difference between the minimum and maximum z-value of all points

$$\delta=|\max(G[3,:])-\min(G[3,:])| \tag{7}$$

The steepness is obtained by calculating the angle between a plane-fit through the points and a flat reference plane. The mean value is subtracted from each of the three dimensions.

$$G_O=G-\mathrm{mean}(G) \tag{8}$$

The normal vector of the plane fitted through these points is the third column of the transposed right eigenvectors from the singular value decomposition.

$$U\Sigma V^T=G_O \tag{9}$$

$$n_p=V^T[3] \tag{10}$$

V is defined as $(v1,v2,v3)$. $v1$ and $v2$ extends across the collection of 3D points, which approximate best the fitted plane. $v3$ is associated to the third singular value $\sigma=0$ and is not included in that plane, but normal to it. In our definition we ordered the singular values in decreasing order.

If the normal vector points into the $-z$ direction, it is reversed to ensure that the angle is between $-90°$ and $90°$. A reference normal vector n_r, the normal vector of the x-y-plane, is determined, which points in the direction $[001]^T$. The angle β is the magnitude of the highest inclination:

$$\beta=\arccos\left(\frac{|n_r*n_p|}{\sqrt{n_r{}^2}\sqrt{n_p{}^2}}\right) \tag{11}$$

Please note that Equation 11 computes the angle between the normal of the reference plane and the normal of fitted plane. The fitted plane is the plane which best approximates all points in one grid cell. The angle between the normals is computed via the dot product and corresponds to the magnitude of the inclination.

To smooth local non-linearities in β, values of a lower resolution 60×60cm are also computed. The final value of each 20×20cm cell is the average of that cell's value and the value of a 60×60cm cell centered on that cell, mapped to integers in the range $[0-255]$.

For each metric (ν, δ, β) and for both resolutions two thresholds th_1 and th_2 must be heuristically selected in the statistical method, (i) th_1 to detect outlier values ν and reassign them closer to the rest of the range to prevent compression

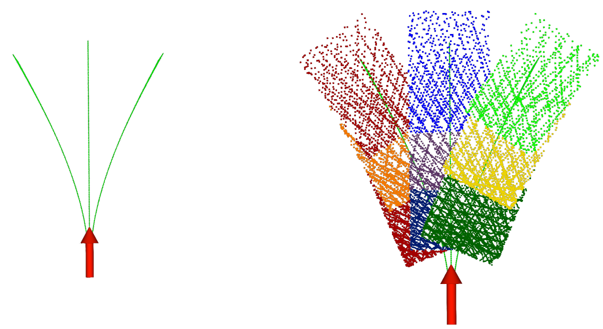


Fig. 7. Local Planning, showing generated trajectories(L). The red arrow symbolizes the current robot position and direction. (R) three patch frames per trajectory are allocated and filled with points from the local point cloud. For autonomous operation 11 trajectories are generated at a angular discretization of 1.7°.

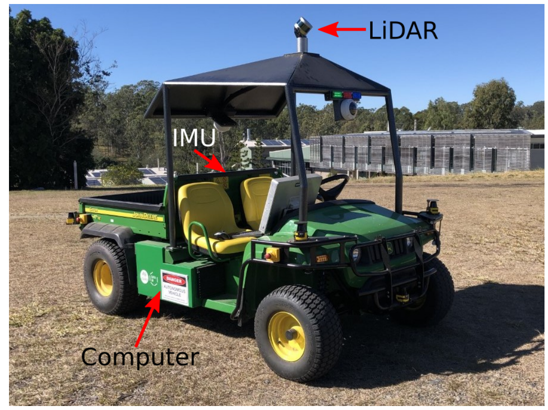


Fig. 8. John Deere TE Gator: The spinning Velodyne PUCK VLP-16 LiDAR mounted on the roof of the vehicle is angled at 45 ° to maximize point coverage. The LiDAR base rotates at approximately 0.5 Hz and LiDAR measurements are streamed in at 20 Hz.

during normalisation, and (ii) th_2 the value above which the cell is allocated as an impassable object.

$$v_i = \min(v_i, th_i) \tag{12}$$

where $i \in \{1,2\}$. These thresholds are terrain-dependent and require tuning.

2) Learning-based Roughness Estimation LRE: The NN is trained to predict the cost of the IMU signal from a point cloud patch, which is a regression problem. Because the patch and the final resolution of the cost map differ in size and shape, the discretized global point cloud is convolved by a filter of the patch's size to extract all points within its boundaries. The resulting patch is down-cast and used by the NN to predict the terrain cost.

As global planning must be pose invariant, this operation is executed 4 times, rotating the filter by 90° each time. During data extraction and NN training, it is assumed that the vehicle always drives forward over a patch. Additionally, one image at one position is input t times into the network because the NN's input is an image time-series. The output is averaged over the t time steps and the 4 rotations.

3) Planning Algorithm: To evaluate the utility of global cost map to perform planning, the baseline Dijkstra algorithm is applied. The vertex cost corresponds to the cells values on the cost map. The edge cost c_e is based on distance and the traversability score, aiming to jointly minimise both terrain cost and the Euclidean distance to the goal. Equation (13) defines c_e when the robot is traveling from state s_1 to state s_2. $D[s]$ is the terrain cost at state s, $d(s_1,s_2)$ is the Euclidean distance between the two states, and e is the user-defined Euclidean distance parameter. If e is zero, the planner takes only the terrain into account, making the global path jittery. On the other hand, if e is 1, the robot acts like on a binary occupancy grid.

$$c_e = \frac{1}{2}(1-e)(D[s_1]+D[s_2]) + e \times d(s_1,s_2) \tag{13}$$

V. Results

In this section we present key results from our experiments. Initially, we describe our experimental setup and test site. Secondly, the accuracy of the learning approach is assessed. Then, the global continuous cost maps are evaluated in field experiments. Finally, use of our techniques as effective real-time local planners are demonstrated.

A. Experimental Setup

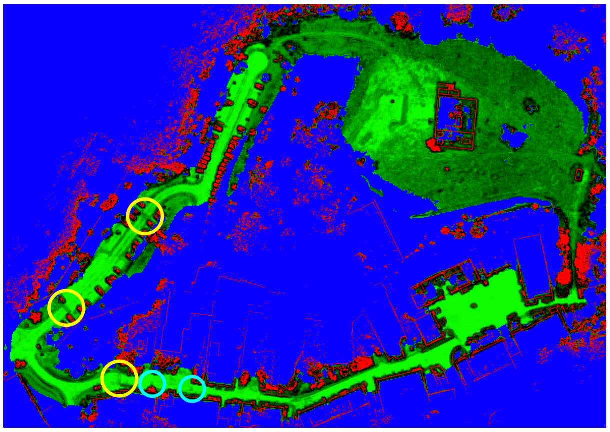
1) Autonomous Ground Vehicle: The AGV used is a John Deere TE Gator equipped with a spinning Velodyne PUCK VLP-16 LiDAR and a Microstrain-CV5 IMU. The vehicle is operated in autonomous mode to ensure a consistent speed throughout the experiments. Posemap [51] and C-SLAM [47],[52] are used for mapping and localisation. The algorithms run on a LGA1151 CPU2.8 GHz and 64 Gb of RAM.

2) Test Site: Our test site (QCAT), located in Brisbane, Australia, contains a large area characterized by a varied mixture of different terrain features (Figure 9(c)). It comprises an urban-like environment, industrial sheds, asphalt roads and a large off-road area consisting of steep and flat grass, dirt, and pebble ground. For global cost maps, we use a pre-computed point cloud of the site. For local planning, we generate and update local maps online during operation.

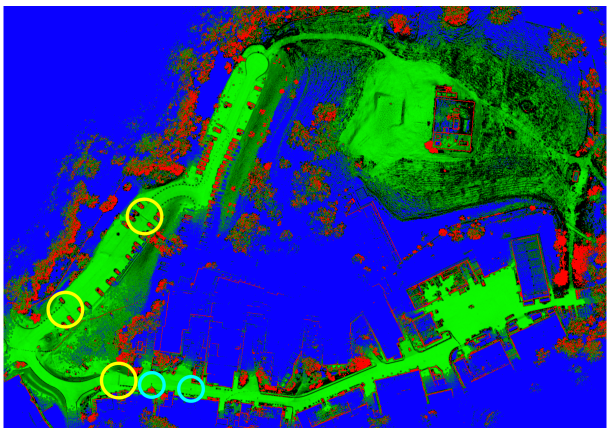
B. Neural Network Accuracy

We initially evaluate the accuracy of the IMU signal prediction with the NN approach. NN performance is based on comparisons between the predicted cost C_p to the ground truth C_r, and the cost signal C. The NN is trained on 119328 images, divided into 90% training data and 10% validation data. NN evaluation is conducted on an additional test set of 12557 terrain images.

The proposed IMU cost C and real IMU cost C_r are defined in Section IV-A1b. The NN's prediction, C_p is shown on the left image in figure 10. The middle image shows C, which is used for network training. On the right image, C_r is displayed, which represents the terrain traversability, which



(a) Colored Learning-Based Global Cost Map.



(b) Colored Statistical-Based Global Cost Map.

(c) Different areas of the QCAT in Brisbane, Australia, highlighted on a satellite image (©Google).

Fig. 9. a) and b) The cost map of the site is shown for both SRE and LRE approaches: obstacles are red, blue is unknown space, and the terrain traversability is colored from green to black. Light green is easily traversable and black very challenging, but still traversable. Speed bumps are circled in yellow and drain holes in cyan. c) Different regions of the QCAT site are highlighted: Blue represents asphalt and green concrete ground. All other parts refer to off-road terrain. The brown-colored area is a soil path through hilly, grassy terrain, which is colored violet. The yellow part expresses a flat, grassy plateau, which encloses a bump colored in black. The most challenging area is a rough, and steep gravel incline, colored in red.

is then averaged over length N=64, giving an IMU signal of 640 ms per image.

During this test run, in the first 30s the vehicle drives on-road, followed by a long off-road part for approximately 250s, before finally driving back onto the road. The off-road part can be easily seen in Figure 10, where the signal C has a larger magnitude and variance. The on-road part of the ride contains three speed bumps at 290s, 310s, and 370s, which can be identified in the three signals. To compare the signals, 10-fold cross-validation R, mean-squared error (MSE), and mean-absolute error (MAE) are presented in Table I. The correlation between the prediction C_p to C is 0.88 and to C_r is 0.71.

TABLE I
COMPARING THE PREDICTED C_p TO THE REAL IMU COST C_r AND TO THE GROUND TRUTH COST COEFFICIENT C. THE CROSS-CORRELATION R, MEAN-SQUARED ERROR (MSE), THE MEAN-ABSOLUTE ERROR (MAE) ARE SHOWN.

	R	MSE	MAE
C	0.88	$3.14 \cdot 10^{-4}$	$1.20 \cdot 10^{-2}$
C_r	0.71	$1.21 \cdot 10^{-3}$	$2.31 \cdot 10^{-2}$

C. Local Cost Map Generation

We now turn our attention to local cost map generation, to be used by a local planner. For each approach (SRE, LRE, BM) 6 experiments are conducted, covering a combined distance of approximately 300 m per approach. The BM only relies on the global path computed on an occupancy grid. /gabrielThe conducted path of the BM is not exactly straight because the steering controller is not precise enough on rough terrain. SRE and LRE choose their trajectory based on the (identical) global path and on locally sensed costs while driving. In this scenario, the robot can periodically deviate from the global path before returning because of local terrain features. At each decision point, the algorithm generates costs for 11 potential trajectories at 1 Hz planning frequency, with the controller running at 20 Hz. Figure 11 shows 3 indicative experiments out of the 6 conducted.

There is no metric that enables direct evaluation of different trajectories, therefore we analyse properties recorded from real-field experiments. Table II presents the measurements summed over all 6 autonomously driven trajectories for each approach. Gradient β, variance ν in z, and residual in z provide roughness and inclination estimates, computed from patch sequences extracted along the driven path. Path length is shorter for the BM. However, the proposed roughness characteristics (β, ν, residual, C) allow for a much smoother ride for SRE and LRE. Despite longer paths for SRE and LRE, the energy and Cost of Transport (COT) is notably lower. COT is a commonly-used measure of locomotion efficiency, and defined as COT$= E/gmd$, where g is the gravitational constant, m the mass of the vehicle (750 kg, in our test vehicle), and the d the distance travelled. For the sake of completeness, trajectory curvature is also recorded. We see a strong pattern of the SRE and LRE permitting a smoother, flatter, and more energy-efficient strategy for local navigation.

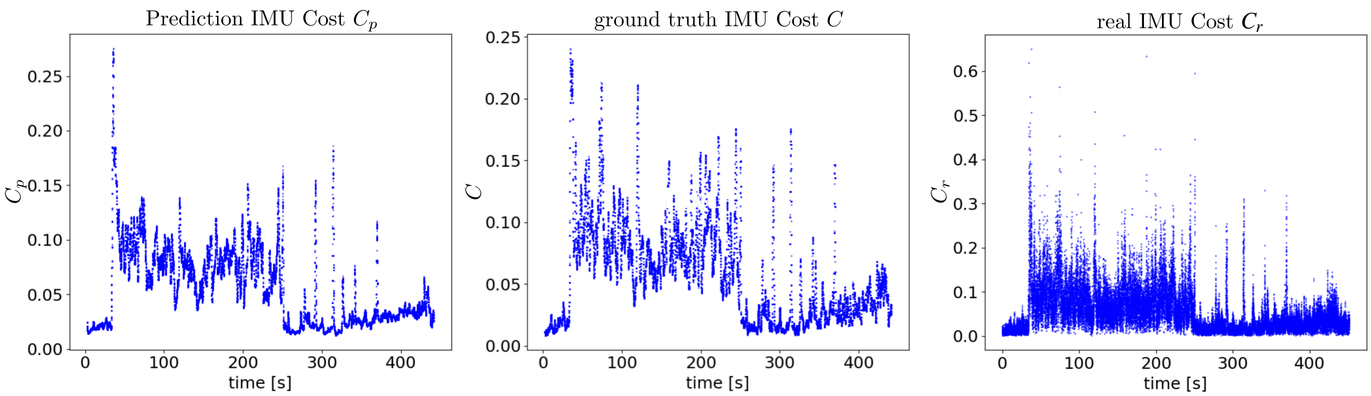
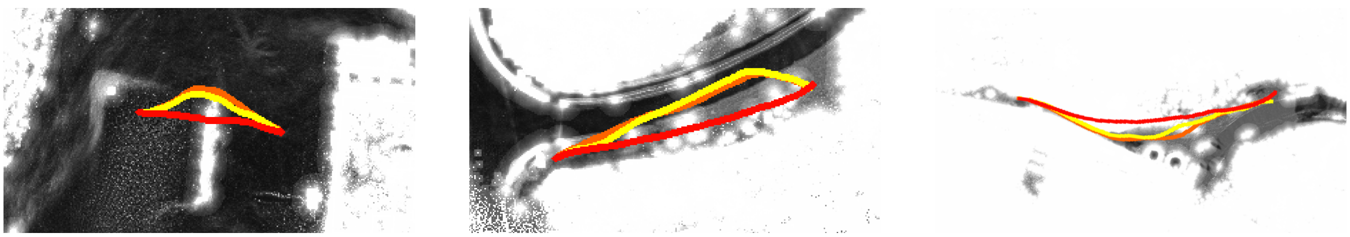


Fig. 10. Showing L-R: predicted cost C_p, ground truth C, cost signal C_r. Between seconds 30 and 250 the vehicle drives on off-road, then transitions on to a road segment with three speed bumps, at seconds 290, 310, and 370 respectively.



(a) Bump trajectories. (b) Street vs. off-road trajectories. (c) High vs. low grass trajectories.

(d) Bump image. (e) Street vs. off-road image. (f) High vs. low grass image.

Fig. 11. Local Planning: Figures 11(a), 11(b), 11(c) show the autonomously driven trajectories for each planning technique. All trajectories are driven from left to right. The trajectory driven using the binary method is illustrated in red, SRE in yellow, and LRE in orange. They are plotted on the continuous global cost map, where brighter pixels correspond to higher cost. Figures 11(d), 11(e), 11(f) show the view recorded from the start point of each path. Figure 11(d) depicts a 30 cm high and a 8 m long bump surrounded by flat, grassy terrain. Figure 11(e) shows a paved road adjacent to a relatively flat grassy surface, where the start and end state of the path are located and Figure 11(f) depicts high and low grass. Please note that in Figure 11(c) the path of the BM is slightly curved because the grass on the left side is considered as full obstacle (completely untraversable) as it is too high.

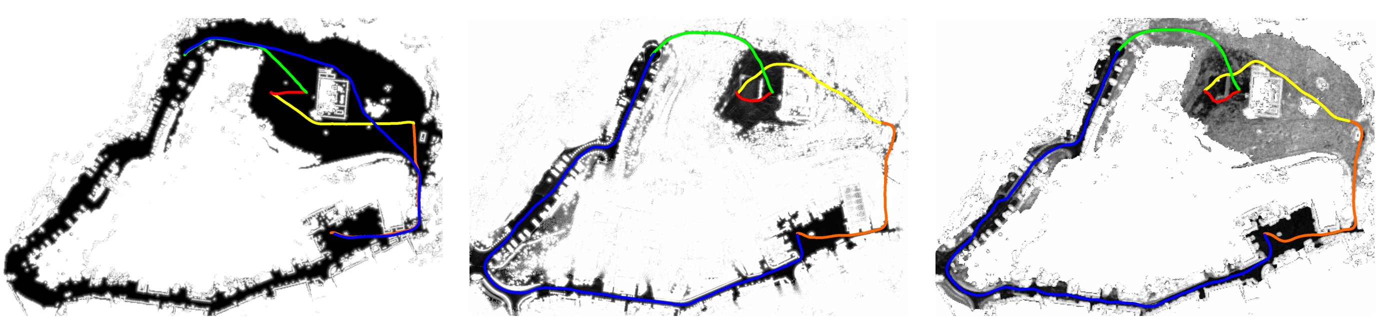


Fig. 12. Five different paths from the binary occupancy grid (left), statistical map (middle), and learning-based map (right). Each color represents a different path with identical start and endpoints. Differences between the continuous cost maps and binary map are visually evident.

TABLE II
PATH CHARACTERISTICS FOR EACH LOCAL PLANNING METHOD.

Characteristics	Method BM	SRE	LRE
Length [m]	**288.4**	296.1	297.2
Curvature $[1/m]$	**0.75**	0.97	0.94
Gradient β [°]	4996	2732	**2313**
Variance ν in z	5.89	2.43	**1.15**
Residual in z	347927	217422	**168335**
IMU cost C	81.3	69.0	**65.5**
Energy [kWh]	0.42	0.41	**0.40**
COT	1.51	1.44	**1.39**

D. Global Terrain Maps

Figure 9(a) shows the learning-based cost map of the site. Red pixels represent non-traversable obstacles. Blue pixels do not contain enough LIDAR points and are marked as unknown. The green color channel represents terrain difficulty and varies from easily traversable (light green) to harder to traverse (dark green).

Comparing different cost maps is a rather difficult task, as there is no metric that enables direct evaluation. We therefore conduct the same path-based analysis as for the local case. The occupancy grid represents the environment as obstacles or free-space.

For each approach, 5 paths covering a total distance of $\approx$ 1km are planned and autonomously driven on the Gator (Figure 12), which follows the global paths as closely as possible as no local costs are considered. For the paths on each map, several characteristics are calculated to quantify the terrain smoothness of the paths and thus the accuracy of the terrain analysis.

TABLE III
FOR EACH OF THE MAPS IN FIGURE 12, SEVERAL PATH CHARACTERISTICS ARE COMPUTED FOR THE 5 PATHS SHOWN IN THAT FIGURE. THE RESULTS SHOWN IN THIS TABLE ARE THE AVERAGE RESULTS FOR ALL PATHS.

Characteristics	BM	SRE	LRE
Length [m]	**774**	980	983
Planning Time [s]	**50.91**	54.12	56.86
Curvature $[1/m]$	**0.17**	0.19	0.24
Gradient [°]	5902	**3759**	3976
Variance in z	7.63	**2.43**	3.02
Residual in z	595358	**376595**	399331
IMU cost C	259	183	**180**
Energy [kWh]	**0.353**	0.363	0.357
COT	0.223	0.181	**0.178**

Table III presents the measurements averaged over all 5 autonomously driven paths for each map. The path length is shorter for the binary map and therefore the energy is lower. However, terrain roughness characteristics (β, ν in z, residual in z, IMU cost C) reveal a much smoother ride for the two continuous maps and are similar for both approaches. The results are inherent to what the methods were trying to minimise. SRE selects paths only dependent on point cloud data and therefore shows better results on them. LRE performs better on the IMU cost on which it was trained initially. As with local planning, and despite longer paths, the COT is lower for SRE and LRE. For the sake of completeness, planning time and path curvature are also provided. The planning time does not vary significantly between the three approaches. On the binary map, the planner searches in each direction simultaneously, whereas on the terrain map, the planner searches first on 'easy' and later on more difficult terrain. The curvature is lower for paths computed on the binary map. Continuous maps enhance the path selection beyond sensor range allowing for a smoother, flatter, and more energy-efficient path.

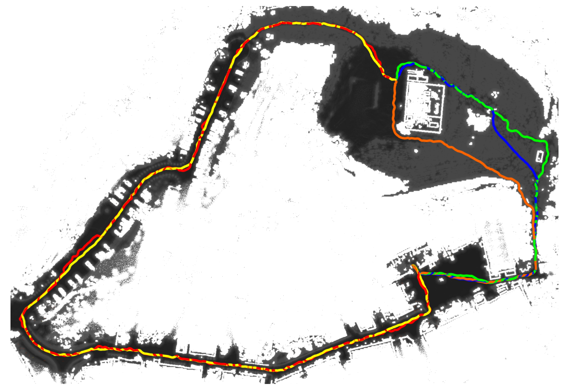


Fig. 13. Computed paths in simulation dependent on the Euclidean distance parameter. Each color represents a different path with identical start and endpoints using different parameters for e. A lower value can result in a shorter but rougher path, whereas a higher value can yield to a longer, but smoother path.

To balance between the terrain traversability cost and the Euclidean distance cost, the weight e is used, as explained in equation 13. Results are illustrated in Figure 13, showing computed paths that are dependent on the Euclidean distance parameter. To assess the sensitivity of this parameter, identical waypoints are given to the planner, each time with a different value of e (ranging from 0.0 to 0.5 in increments of 0.05). In orange, e is 0.5 heavily favours shortest distance. Blue has a e=0.3, and green e=0.2. Both elect to follow a rough, steep, and narrow back-road at the right side in the image. Yellow has e=0.1, and red e=0.0. The majority of these paths follow the road. Yellow relies mainly on the terrain and even on-road generates a jittery path. The best parameter setting depends on use preference; as a guide the experiments in section V-D use e=0.15.

A video illustrating the algorithms and the vehicle operating autonomously is shown in https://youtu.be/2PGWs27XlsU.

VI. CONCLUSION

This work illustrates use of continuous cost maps to represent terrain traversability. This information is used for both local and global planning. The goal is to achieve safer, smoother, and energy efficient operations.

We applied two different methods for the generation of the cost map: a statistical approach, and a learning-based approach, and provided a traditional binary cost map as a comparison baseline. The former estimates the terrain based on statistical properties computed from the point cloud. For the latter, we developed an entire learning pipeline and proposed a novel definition of terrain roughness by sensing the ground with an IMU sensor, and predicting an *IMU cost* solely from a point cloud patch using a CNN-LSTM NN. Each approach was evaluated in a variety of autonomous driving experiments, with features such as path smoothness and Cost of Transport assessed. Results show that both continuous approaches are general enough to be successfully applied for either local or global planning. The learning approach does not assume useful terrain features as strongly as the statistical approach, and does not require tuning of the thresholds on the cost map, which increases generality further. The learning-based approach either meets or exceeds the performance of the other approaches in most cases. Overall, both approaches outperform an occupancy grid by a considerable margin, with the learning approach performing slightly better. During local planning experiments, we observe the local planner replanning due to sensed terrain features (i.e., not obstacles), due to the directionality afforded to patch costing in the local case. Both continuous approaches obtain similar performance and yield to much better results than a trajectory only relying on the Euclidean distance to the goal. Further work will include incorporating colour information by giving the NN a prior traversability class based on semantic segmentation. Also, feeding a colorized point cloud into the NN, while keeping the architecture identical, can provide extra sources of data to the network to potentially improve performance.

REFERENCES

[1] W. Xi, Y. Ou, J. Peng, and G. Yu, "A new method for indoor low-cost mobile robot SLAM," in 2017 IEEE International Conference on Information and Automation, ICIA 2017, 2017.

[2] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y. W. Seo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, "Autonomous driving in Urban environments: Boss and the Urban Challenge," in Springer Tracts in Advanced Robotics, 2009.

[3] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," IEEE Transactions on Intelligent Vehicles, 2016.

[4] X. Meng, Z. Cao, S. Liang, L. Pang, S. Wang, and C. Zhou, "A terrain description method for traversability analysis based on elevation grid map," International Journal of Advanced Robotic Systems, 2018.

[5] D. Langer, J. Rosenblatt, and M. Hebert, "A Behavior-Based System for Off-Road Navigation," IEEE Transactions on Robotics and Automation, 1994.

[6] 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," Proceedings-IEEE International Conference on Robotics and Automation, 2000.

[7] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L. E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Stanley: The robot that won the DARPA Grand Challenge," Springer Tracts in Advanced Robotics, 2007.

[8] D. Helmick, A. Angelova, and L. Matthies, "Terrain adaptive navigation for planetary rovers," Journal of Field Robotics, 2009.

[9] A. Talukder, R. Manduchi, A. Rankin, and L. Matthies, "Fast and reliable obstacle detection and segmentation for cross-country navigation," 2003.

[10] C. J. Holder, T. P. Breckon, and X. Wei, "From on-road to off: Transfer learning within a deep convolutional neural network for segmentation and classification of off-road scenes," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2016.

[11] A. Valada, J. Vertens, A. Dhall, and W. Burgard, "AdapNet: Adaptive semantic segmentation in adverse environmental conditions," in 2017 IEEE International Conference on Robotics and Automation (ICRA), 5 2017, pp. 4644–4651.

[12] N. Hirose, A. Sadeghian, M. Vázquez, P. Goebel, and S. Savarese, "GONet: {A} Semi-Supervised Deep Learning Approach For Traversability Estimation," CoRR, vol. abs/1803.0, 2018. [Online]. Available: http://arxiv.org/abs/1803.03254

[13] A. Angelova, L. Matthies, D. Helmick, and P. Perona, "Slip prediction using visual information," in Robotics: Science and Systems, 2007.

[14] M. Bajracharya, B. Tang, A. Howard, M. Turmon, and L. Matthies, "Learning long-range terrain classification for autonomous navigation," in Proceedings - IEEE International Conference on Robotics and Automation, 2008.

[15] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine, "Self-Supervised Deep Reinforcement Learning with Generalized Computation Graphs for Robot Navigation," in Proceedings - IEEE International Conference on Robotics and Automation, 2018.

[16] D. Silver, J. A. Bagnell, and A. Stentz, "Learning from demonstration for autonomous navigation in complex unstructured terrain," International Journal of Robotics Research, vol. 29, no. 12, pp. 1565–1592, 2010.

[17] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," IEEE Robotics and Automation Magazine, 1997.

[18] S. Thrun, "Robotic Mapping: A Survey," Science, 2002.

[19] A. Elfes, "Sonar-Based Real-World Mapping and Navigation," IEEE Journal on Robotics and Automation, 1987.

[20] B. Kuipers and Y. T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," Robotics and Autonomous Systems, 1991.

[21] D. Kortenkamp and T. Weymouth, "Topological mapping for mobile robots using a combination of sonar and vision sensing," in Proceedings of the National Conference on Artificial Intelligence, 1994.

[22] G. Ishigami, A. Miwa, K. Nagatani, and K. Yoshida, "Terramechanics-based model for steering maneuver of planetary exploration rovers on loose soil," Journal of Field robotics, vol. 24, no. 3, pp. 233–250, 2007.

[23] ——, "Terramechanics-based analysis on slope traversability for a planetary exploration rover," in Proceedings of the International Symposium on Space Technology and Science, vol. 25, 2006, p. 1025.

[24] D. B. Gennery, "Traversability analysis and path planning for a planetary rover," Autonomous Robots, vol. 6, no. 2, pp. 131–146, 1999.

[25] B. Hamner, S. Singh, S. Roth, and T. Takahashi, "An efficient system for combined route traversal and collision avoidance," Autonomous Robots, vol. 24, no. 4, pp. 365–385, 2008.

[26] M. G. Bekker, "Introduction to terrain-vehicle systems. part i: The terrain. part ii: The vehicle," MICHIGAN UNIV ANN ARBOR, Tech. Rep., 1969.

[27] P. Krüsi, P. Furgale, M. Bosse, and R. Siegwart, "Driving on Point Clouds: Motion Planning, Trajectory Optimization, and Terrain Assessment in Generic Nonplanar Environments," Journal of Field Robotics, vol. 34, no. 5, 2017.

[28] J. F. Lalonde, N. Vandapel, D. F. Huber, and M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility," Journal of Field Robotics, 2006.

[29] K. Iagnemma, F. Genot, and S. Dubowsky, "Rapid physics-based rough-terrain rover planning with sensor and control uncertainty," Proceedings - IEEE International Conference on Robotics and Automation, 1999.

[30] H. Seraji and A. Howard, "Behavior-based robot navigation on challenging terrain: A fuzzy logic approach," IEEE Transactions on Robotics and Automation, 2002.

[31] Y. Tanaka, Y. Ji, A. Yamashita, and H. Asama, "Fuzzy based traversability analysis for a mobile robot on rough terrain," in Proceedings - IEEE International Conference on Robotics and Automation, 2015.

[32] A. Krebs, C. Pradalier, and R. Siegwart, "Comparison of Boosting Based Terrain Classification Using Proprioceptive and Exteroceptive Data," in Springer Tracts in Advanced Robotics, 2009.

[33] F. G. Oliveira, E. R. Santos, A. A. Neto, M. F. Campos, and D. G. Macharet, "Speed-invariant terrain roughness classification and control based on inertial sensors," in Proceedings - 2017 LARS 14th Latin American Robotics Symposium and 2017 5th SBR Brazilian Symposium on Robotics, LARS-SBR 2017 - Part of the Robotics Conference 2017, 2017.

[34] V. Surblys, V. Žuraulis, and E. Sokolovskij, "Estimation of road roughness from data of on-vehicle mounted sensors," Eksploatacja i Niezawodnosc - Maintenance and Reliability, vol. 19, pp. 369–374, 2017.

[35] W. Wen, "Road Roughness Detection by Analysing IMU Data," 2008.

[36] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "SEGCloud: Semantic segmentation of 3D point clouds," in Proceedings - 2017 International Conference on 3D Vision, 3DV 2017, 2018.
[37] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel, "On the segmentation of 3D lidar point clouds," in Proceedings - IEEE International Conference on Robotics and Automation, 2011.
[38] M. Kragh, R. N. Jørgensen, and H. Pedersen, "Object detection and terrain classification in agricultural fields using 3d lidar data," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2015.
[39] A. Santamaria-Navarro, E. H. Teniente, M. Morta, and J. Andrade-Cetto, "Terrain classification in complex three-dimensional outdoor environments," Journal of Field Robotics, 2015.
[40] B. Suger, B. Steder, and W. Burgard, "Traversability analysis for mobile robots in outdoor environments: A semi-supervised learning approach based on 3D-lidar data," in Proceedings - IEEE International Conference on Robotics and Automation, 2015.
[41] J. L. Martínez, M. Morán, J. Morales, A. Robles, and M. Sánchez, "Supervised learning of natural-terrain traversability with synthetic 3D laser scans," Applied Sciences (Switzerland), 2020.
[42] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in Advances in Neural Information Processing Systems, 2017.
[43] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017.
[44] H. Thomas, C. R. Qi, J. E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in Proceedings of the IEEE International Conference on Computer Vision, 2019.
[45] F. Lomio, E. Skenderi, D. Mohamadi, J. Collin, R. Ghabcheloo, and H. Huttunen, "Surface Type Classification for Autonomous Robot Indoor Navigation," CoRR, vol. abs/1905.0, 2019. [Online]. Available: http://arxiv.org/abs/1905.00252
[46] F. G. Oliveira, A. A. Neto, D. Howard, P. Borges, M. F. Campos, and D. G. Macharet, "Three-dimensional mapping with augmented navigation cost through deep learning," Journal of Intelligent & Robotic Systems, vol. 101, no. 3, pp. 1–21, 2021.
[47] M. Bosse and R. Zlot, "Continuous 3D scan-matching with a spinning 2D laser," 2009.
[48] R. Zlot and M. Bosse, "Efficient large-scale three-dimensional mobile mapping for underground mines," in Journal of Field Robotics, 2014.
[49] Y. Zhou and K. Hauser, "Incorporating side-channel information into convolutional neural networks for robotic tasks," in Proceedings - IEEE International Conference on Robotics and Automation, 2017.
[50] J. Hwang, J. Kim, A. Ahmadi, M. Choi, and J. Tani, "Predictive coding-based deep dynamic neural network for visuomotor learning," in 7th Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics, ICDL-EpiRob 2017, 2018.
[51] P. Egger, P. V. Borges, G. Catt, A. Pfrunder, R. Siegwart, and R. Dube, "PoseMap: Lifelong, Multi-Environment 3D LiDAR Localization," in IEEE International Conference on Intelligent Robots and Systems, 2018.
[52] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a spring-mounted 3-D range sensor with application to mobile mapping," IEEE Transactions on Robotics, 2012.

Gabriel Günter Waibel Gabriel received his BSc. in 2018 and MSc. in 2020 in Information Technology and Electrical Engineering from ETH Zurich, Switzerland. He conducted his master's thesis at the Robotics and Autonomous System Group, CSIRO, Brisbane. Currently, he is a research engineer at the Robotic Systems Lab, ETH Zurich. There he was part of Team Cerberus that won the DARPA SubT Final Event and is participating in the ESA Space Resource Challenge. His research interests lie in the fields of lidar-based localization and mapping, and navigation for robotic systems.

Tobias Löw received his BSc. and MSc. in Mechanical Engineering from ETH Zürich, Switzerland, in 2018 and 2020, respectively. He conducted his master's thesis at the Robotics and Autonomous Systems Group, CSIRO, Brisbane. Currently, he is a Ph.D. student at École Polytechnique Fédérale de Lausanne (EPFL), working in the Robot Learning and Interaction Group at the Idiap Research Institute. His research interests lie in exploiting geometric algebra and tensor decomposition methods for robot motion planning and exploration.

Mathieu Nass received the BSc. in Applied Physics and a MSc. in Computer Science from the University of Twente, the Netherlands, in 2018 and 2020 respectively. During his MSc. both his internship and final thesis were related to SLAM. The latter of which was conducted at DEMCON Enschede, the Netherlands, on integrating Wi-Fi measurements in a graph-based SLAM algorithm. The former was conducted at the Robotics and Autonomous System Group at CSIRO in collaboration with the authors of this paper.

David Howard received the B.Sc. in Computing and M.Sc. in Cognitive Systems from the University of Leeds, UK, in 2005 and 2006 respectively. In 2011 he received the Ph.D. degree from University of the West of England, UK, where he stayed as a Postdoctoral Fellow until 2013. Since 2013, he has been in the Robotics and Autonomous Group at CSIRO in Brisbane, Australia, where he is a Senior Research Scientist and Team Leader. His research interests span machine learning, evolutionary computing, field robotics, and soft robotics. Dr. Howard holds an Adjunct position at the University of Queensland, Australia.

Tirthankar Bandyopadhyay Tirthankar Bandyopadhyay received his PhD in Mechanical Engineering specializing in Robotics in 2010 from the National University of Singapore (NUS). From 2010 to 2013, he was a post-doc researcher in Singapore MIT Alliance for Research and Technology (SMART). Since 2013, he has been with the Robotics and Autonomous Systems Group at CSIRO in Brisbane, Australia, where he is currently a Senior Research Scientist. His research focuses on robot locomotion, motion planning under uncertainty and robot-world interaction.

Paulo Vinicius Koerich Borges received the B.E. and M.Sc. degrees in electrical engineering from Federal University of Santa Catarina (UFSC), Brazil, in 2002 and 2004, respectively. In 2007 he received his Ph.D. degree from Queen Mary, University of London (QMUL). From 2007 to 2008, he was as post-doc researcher at QMUL. Since 2009, he has been in the Robotics and Autonomous Systems Group at CSIRO in Brisbane, Australia, where he is currently a Principal Research Scientist and leads the Robotics Perception Team. His research focuses on visual-based robot localisation, obstacle detection, and multi-sensor information fusion. In 2012-13, he held a visiting scientist appointment at ETH Zurich in Switzerland. Dr. Borges is also an Adjunct Associate Professor with the School of Information Technology at Griffith University, Australia.