Traversability Estimation for Ground Robots

Anonymous Author(s)
Affiliation
Address
email

Abstract: Mobile robots operating on unstructured terrain must predict which areas of the environment they are able to pass in order to plan feasible paths and to react to unforeseen terrain patterns. We address traversability estimation as a heightmap classification problem: we build a convolutional neural network that, given an image representing the heightmap of a patch of terrain, predicts whether the robot will be able to traverse such patch from left to right. The classifier is trained for a specific wheeled robot model (but may implement any other locomotion type such as tracked, legged, snake-like), using simulation data on a variety of procedurally generated training terrains; once trained, the classifier can be quickly applied to patches extracted from unseen large heightmaps, in multiple orientations, thus building oriented traversability maps. We quantitatively validate the approach on real-elevation datasets and implement a path planning approach that employs our traversability estimation.

Keywords: traversability estimation, convolutional neural networks

1 Introduction

In most indoor scenarios, the environment is trivially partitioned in traversable areas (clear floor) and known obstacles (i.e. walls, objects, and other areas through which the robot can't pass); once traversable and non-traversable areas are known, path planning (an essential task for mobile robots) is solved using well-known algorithms [1].

In scenarios with uneven terrain, such as outdoors, segmenting the environment in traversable and non-traversable areas is not as trivial. For example, some areas may be traversable only in a few specific directions: a wheeled robot with limited power could be able to descend but not to ascend a steep slope; a powerful, long and narrow robot with an high center of mass could be able to traverse the same environment both uphill and downhill, but may capsize if traversing perpendicularly to the steepest direction; a bicycle can hop up a side-walk as long as it is proceeding more or less perpendicular to the step, but will crash when approaching the step from an oblique angle. Moreover, criteria for traversability may not be intuitive or easy to model a-priori: a legged robot may be able to negotiate very challenging terrain but get stuck on flat ground with deep holes of comparable size as its feet; vacuum cleaner robots get stuck over power cords laying on the ground; a car with a low clearance may get stuck on a speed bump.

We consider the problem of estimating where and in which directions a given 3D terrain is locally traversable by a specific ground robot, using a general approach based on machine learning that applies regardless on the robots locomotion method (wheeled, tracked, legged, snake-like), physical characteristics (size, motor torque), and low-level controller (anti-skid algorithms for wheeled robots, foothold selection and gait selection algorithms for legged robots). In the following we define that a given portion of a terrain is *locally traversable* in a given direction if the robot, placed in such position and direction, can proceed straight for at least a short distance when driven by its low-level control algorithm.

When robot control parameters are known, traversability is only affected by the characteristics of the terrain around the robot's position. Therefore, for a given robot pose X^{robot} with position p and orientation θ, we consider as input an heightmap patch centered in p and rotated in such a way that the robot is pointing towards the top of the patch. This information is represented as a heightmap

Submitted to the 1st Conference on Robot Learning (CoRL 2017). Do not distribute.

image where pixels indicate height values. Our task is cast as a binary image classification problem (with classes *traversable* vs *non-traversable*), and solved by training a simple convolutional neural network.

Training data is generated by simulating the robot on many procedurally generated training terrains, which represent a variety of obstacles such as ramps, steps, bumps, holes and rugged areas; during such simulations, the robot is spawned in random positions and orientations, and instructed to proceed straight ahead; its progress is monitored and instances for traversable (where the robot successfully proceeds) and non-traversable (where the robot can't proceed) heightmap patches are continuously recorded. Once the model is learned from such training data, it can be applied densely on any large unseen heightmap. Because accurate physical simulation is expensive, evaluating the classifier on many points and orientations of a test terrain is orders of magnitude faster than simulating the robot on these poses.

The **main contribution** of this work relies on the proposal of a complete framework for traversability estimation detailed in Section 3. It includes i) a procedural generation of heightmaps, ii) a novel approach for elevation-map based classification, iii) an orientation-based traversability representation, and iv) an application of such representation on path planning in real elevation maps and with a real robot. Experimental validation and results are described in Section 4 [1]. Limitations and extensions of our approach are discussed in Section 5.

2 Related Work

Estimating traversability is a fundamental capability for many animals and for autonomous mobile robots, because most of their actions depend on their mobility. Traversability can be seen as an *affordance* of the environment on the robot, or vice-versa [2], the former defines which terrain characteristics allow the robot to traverse, the later includes robot's capabilities (e.g. morphology, motor, sensing) to traverse the environment. In robotics, using a simplified (possibly learned) model to estimate traversability is a common approach, because modeling the terrain, the robot and their interaction accurately is difficult and expensive. Several approaches have been proposed to measure traversability and to gather training data [3]. For instance, a robot may label a terrain as difficult to traverse when sensing excessive vibrations [4, 5]; human experts can provide clues like preferred paths in a given terrain that avoid possible non-traversable regions [6]; the robot may learn a traversability classifier on the spot, directly from experience [7].

The outdoor robotics community has researched traversability estimation from a variety of sensing sources, such as stereo cameras [8], laser scanners [9], and time of flight cameras [10]. A common approach to estimate traversability consists of two phases. First, from local sensory data, an elevation map is derived [11]. Then, traversability estimates of parts of the elevation map are computed from geometrical features like *slope*, *roughness*, and *step height* using pre-modeled functions specific to the robot's locomotion and size [12].

Another major approach relies on the use of visual texture to classify the terrain type, e.g. discriminating between rock, sand, and grass; then it derives a traversability score out of the assigned label [4]. Terrain classification is a classical application for supervised machine learning techniques based on generic visual features, transformation-invariant descriptors, texture features, or convolutional neural networks (CNNs) [13, 14], which learn to extract problem-specific visual features. DARPA's project *Learning Applied to Ground Robots project* [15] has advanced learning techniques for terrain traversability classification, including some applications of deep learning [16].

3 Traversability Estimation Framework

Figure 1 illustrates the proposed framework for traversability estimation. Next, we describe how the simulation environment is set-up (Section 3.1), detail the process for generating training and evaluation datasets (Section 3.2), and finally describe classification approaches (Section 3.3).

[1] Data and code to reproduce our results, video demonstrations, and additional information are anonymously available online: `https://github.com/romarcg/traversability_estimation_pioneer`

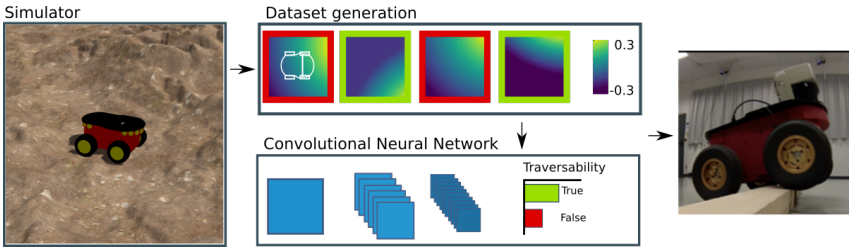

Figure 1: Robot model runs in simulation on procedurally generated terrains (left) to generate datasets linking heightmap patches with their traversability (top); on these datasets we train and evaluate classifiers to estimate the probability that a given heightmap patch is traversable or not (bottom). The learned classifier correctly predicts terrain traversability for a real robot (right).

3.1 Traversability from Simulation

We employ the Gazebo simulator and the ODE physics engine [17] for accurate physical simulation. We simulate the Pioneer 3-AT robot platform of size $49 \times 50 \times 29$ cm (illustrated in Figure 1) moving forward at constant velocity of $0.15m/s$ on an uneven terrain whose shape is determined by a heightmap. The robot's trajectory on the terrain is captured to extract traversability informations.

3.1.1 Heightmap Generation

In order to generate meaningful training data, we need to simulate the robot moving on interesting terrains that pose varied and representative challenges. This could be achieved by using real heightmap data, by creating interesting terrains manually, and/or by synthesizing them using procedural generation techniques. We follow exclusively the third option. Each of our training terrains is generated as a grayscale heightmap, based on summing multiple realizations of random 2D simplex noise [18, 19]: a variant of Perlin noise [20], conceptually similar to it, frequently adopted in the procedural generation literature [21]) with different periods.

For example, a heightmap, corresponding to simplex noise with period 30 cm and scaled such that it extends to a height of 20 cm, models a terrain with medium-sized smooth rocks while a period of 10 m with a 3 m height range yields a landscape with small steep hills. By summing the two heightmaps, one obtains rocky hills. If the *rocks* heightmap is multiplied by a value between 0 and 1 which increases on the x coordinate, one gets flat hills for low values of x which becomes rocky hills for high values of x. Additional features such as holes, bumps protruding from a flat surface, rail-like indentations, steps, and others, can be generated by applying various functions on each value in the heightmap. Some example training terrains are represented in Figure 2. The size of each generated heightmap is 512×512 px that, when simulated, are scaled to represent surfaces of 10×10 m (≈ 2 cm/px resolution).In the supplementary material we provide source code for generating our training dataset and an appendix describing the approach in more detail.

3.1.2 Simulation Process

Once the simulation is initialized with a generated heightmap, the robot is set to a random pose on the map and moves forward at constant velocity without steering. After the robot reaches the edge of the map or gets stuck for some time, it is re-spawned to a different pose to generate a new trajectory.

For each trajectory, the robot poses and their corresponding heightmap patches are recorded; the heightmap patch associated to a pose is centred on the robot's position and oriented in such a way that the robot is facing towards the right of the patch (see Fig. 1-right). If and only if the distance between the current pose $X^{\text{robot}}(t)$ and a future pose $X^{\text{robot}}(t+T)$ is greater than a threshold d and aligned with the robot's orientation, then the patch is labelled as traversable.

Figure 3-left illustrates the traversability labelling for patches along a trajectory with $T = 1$ s and $d = 0.12$ m. Following this trajectory the robot is able to traverse a set of downward-sloped patches until it gets stopped by a bump almost as high as the robot itself. This behaviour is represented in Figure 3-right, where traversable patches are marked with green frames.

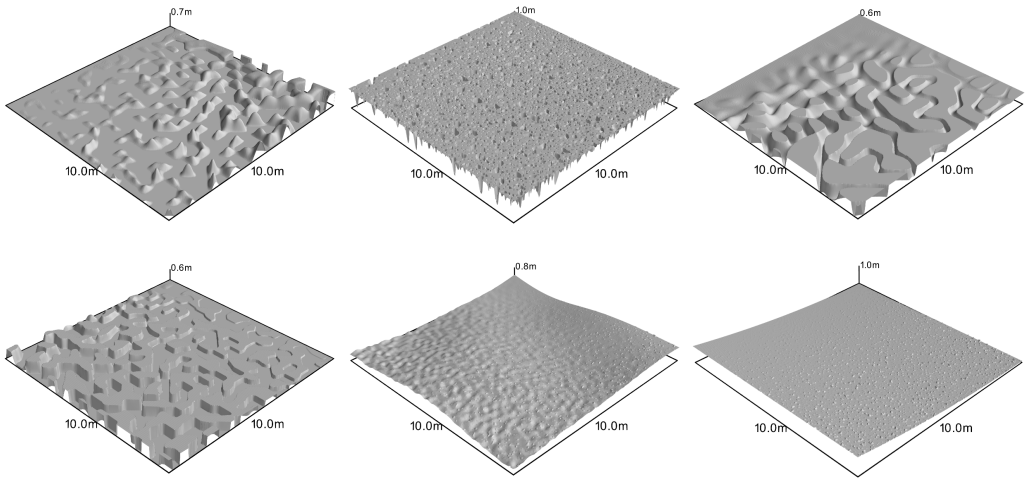

Figure 2: Examples of heightmaps for generating traversability datasets. These maps includes common terrain varieties as: bumps, holes, rails, steps, and slopes.

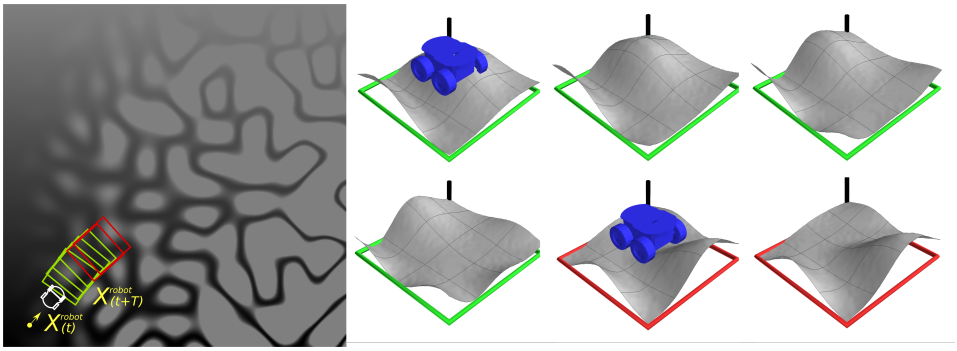

Figure 3: Trajectory extracted from a synthetic dataset (left). Robot silhouette and yellow arrow indicates the initial pose for the corresponding trajectory. Surfaces of the patches from the same trajectory (right). Green frames represent patches labelled as traversable, red frames as non-traversable.

3.2 Dataset Generation

From each simulated trajectory (on average 20 s), we sample the robot pose and the corresponding heightmap patch at 20 Hz. The patch and its traversability label represents an instance in the dataset.

We generate three datasets: D_{train}, a training dataset (450 k samples) from procedurally generated heightmaps (see Section 3.1.1); $D_{\text{eval,syn}}$, an evaluation dataset (150 k samples) with synthetic heightmaps also procedurally generated; $D_{\text{eval,real}}$, an evaluation dataset from real elevation maps (60 k samples). The maximum height of the heightmaps is 1 m for D_{train} and $D_{\text{eval,syn}}$; 3 m for $D_{\text{eval,gravelpit}}$. All datasets were generated using the robot described in Sec. 3.1, a patch size of 60×60 px, $T \approx 1$ s and $d = 0.12$ m.

Figure 3-right illustrates patches from the trajectory of Fig. 3-left. The height values in each patch are offset in such a way that the patch center (i.e. at robot's position) is mapped to height 0. This makes the patches independent on their absolute height on the heightmap, a feature that does not affect traversability.

Dataset $D_{\text{eval,gravelpit}}$ is generated using an elevation map from a Swiss gravel pit obtained by a flying drone [22]. The original area of this map is $0.48\ \text{km}^2$, with a resolution of $5\ \text{cm/px}$ and a maximum height of 50m. We cropped and scaled two regions of this map to form two heightmaps of 10×10 m with a resolution of $1.9\ \text{cm/px}$ and a maximum height difference of 3 m. Figure 4 shows a reference image of the gravel pit and the surfaces of two extracted heightmaps.

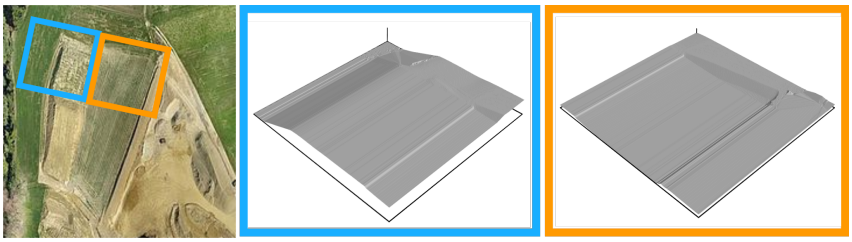

Figure 4: Surfaces of two heightmaps from real elevation data. Heightmaps were extracted from a mapping of a Swiss gravel pit [22]. Coloring represents height.

3.3 Training Traversability Classifiers

We address the problem of estimating terrain traversability as classification on heightmap data. We compare two alternative approaches: extracting descriptive features from each heightmap patch and then applying standard statistical classification techniques [23], or adopting Convolutional Neural Networks, a now-standard deep-learning approach [24] which operates directly on the raw input data. In either case, the output of the classifier indicates whether a patch is traversable.

For the **feature-based approach** we should extract from the input heightmap patch some quantities that provide traversability cues, for example the average terrain steepness in the robot's motion direction (i.e. from left to right of the patch), or the maximum height of any steps in patch. In our case, we compute the Histogram of Gradients (HOG) of the heightmap patch, which includes these pieces of information (e.g., the gradient of a heightmap corresponds to the local steepness). Computing HOG over 6 orientations, 8×8 px per cell, and a block of 3×3 cells, results on a descriptor with 324 features that we classify by means of a Random Forest (RF) classifier [25] with 10 trees.

In the **CNN-based approach**, it is expected that the network autonomously learns meaningful, problem-specific features; because the input data is high-dimensional and no prior knowledge of the problem is provided to the model, this approach requires more training data. Our CNN is built on the Keras [26] frontend powered by TensorFlow [27], and implements a 60×60 px input layer, followed by: a 3×3 convolution layer with 5 output maps; a 3×3 convolution layer with 5 output maps; a 2×2 Max-Pooling layer; a 3×3 convolution layer with 5 output maps; a fully connected layer with 128 output neurons; a fully connected layer with 2 output neurons followed by a softmax layer (output). All layers implement the ReLU activation function. The network is trained for 100 epochs to minimize a categorical cross-entropy loss using the Adadelta optimizer.

4 Experimental Results

The following sections detail the comparison of the two classifiers described in Section 3.3 and an additional baseline dummy classifier that always returns the class most frequent in the training set.

4.1 Classification Results

Performance of the three estimators on the two different evaluation datasets (disjoint from the training set) $D_{\text{eval,syn}}$ and $D_{\text{eval,gravelpit}}$ is presented in Table 1. We observe that the CNN estimator outperforms both the baseline and feature-based approaches on both evaluation datasets. Performance is lower on the $D_{\text{eval,gravelpit}}$ dataset than on $D_{\text{eval,syn}}$, probably because of some elevation patterns, such as narrow and low barriers that may block the robot, which are not represented in procedurally-generated training data. Nonetheless, many features of real terrains such as slopes, rails and bumps are correctly classified.

4.2 Traversability Estimation on Real-world Datasets

We evaluate our trained classifiers on an additional real elevation map. This dataset consists on a mining quarry (see Fig. 5) of 0.51km^2 [22]. The map contains some challenging roads and barriers

	$D_{\text{eval,syn}}$		$D_{\text{eval,gravelpit}}$	
	ACC	AUC	ACC	AUC
CNN	0.9264	0.9709	0.8327	0.8391
Feature-based	0.7031	0.7465	0.7698	0.8022
Baseline	0.5447	0.4971	0.5776	0.5044

Table 1: Performance of CNN, feature-based and baseline approaches. Accuracy (ACC) and area under the ROC curve (AUC) metrics are from a synthetic dataset $D_{\text{eval,syn}}$ generated similarly as the training dataset, and from the gravel pit dataset $D_{\text{eval,gravelpit}}$ described in Fig. 4.

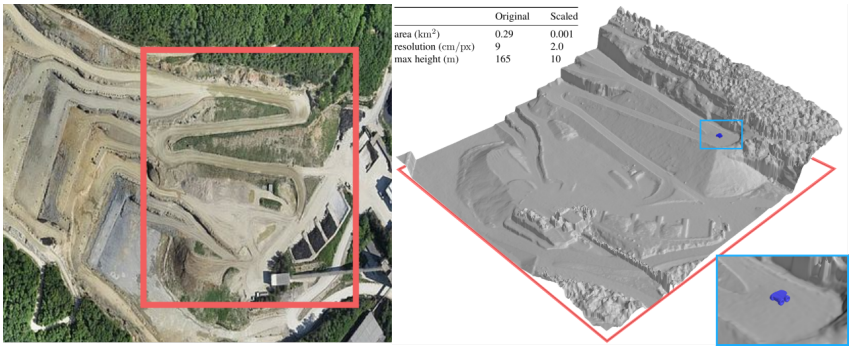

Figure 5: Heightmap extracted from Sensefly's mining quarry dataset [22]. Reference top-view image and perspective view of the elevation map from the extracted red region are shown at left and right respectively. Pioneer 3-AT's model is displayed for size comparison.

designed for cars and trucks. To make the map more suitable for our robot, which is about $0.1\times$ the size of a mining truck, we rescaled the map to $0.05\times$ its original size.

For this analysis, we fix a direction and iterate over the entire heightmap extracting patches of 60×60px with a stride of 5px. This process is equivalent as translating the robot's position over the map while keeping a fixed orientation. Figure 6 shows the traversability estimation for the mining quarry for four orientations, indicated by the arrows. Traversability is represented as a colored overlay on the surface of the heightmap (traversable is green, not traversable is gray). The minimum traversability overlay (blue frame) is calculated as the the minimum traversability estimation at a position from a set of possible orientations (in this case 32).

The estimator correctly marks the main road ($\approx$1.9 m wide) as traversable in all directions. Very steep or vertical slopes are recognized as non-traversable when going up but, dangerously, marked traversable when going down. This particular result is congruent with the definition of traversability we used, which only considers whether the robot can proceed but does not account for its safety. Slopes are classified as traversable always downhill and, sometimes, transversally. Most of the rough surface at the top of the elevation map is correctly found as non-traversable. The minimum traversability representation highlights only areas that the robot can traverse in any of 32 orientations. These, correctly, correspond to roads and flat terrain.

4.3 Path Planning using Probabilistic Traversability Estimations

Traversability maps can be used to plan a path to a goal position which only passes through patches which are traversable in the direction corresponding to the path's local orientation. In the following, we assume that the robot will get stuck (or crash, or end up in an unrecoverable state) if it attempts to pass through a non-traversable patch[2].

We assume that traversability probabilities of non-overlapping patches are independent, i.e. that the traversability along a path is Markovian. This allow us to compute the probability p to traverse a path

[2]this is a pessimistic assumption because this is not always the case: a robot attempting pass through a wall, for example, is unable to proceed but may be capable of turning back and taking another path

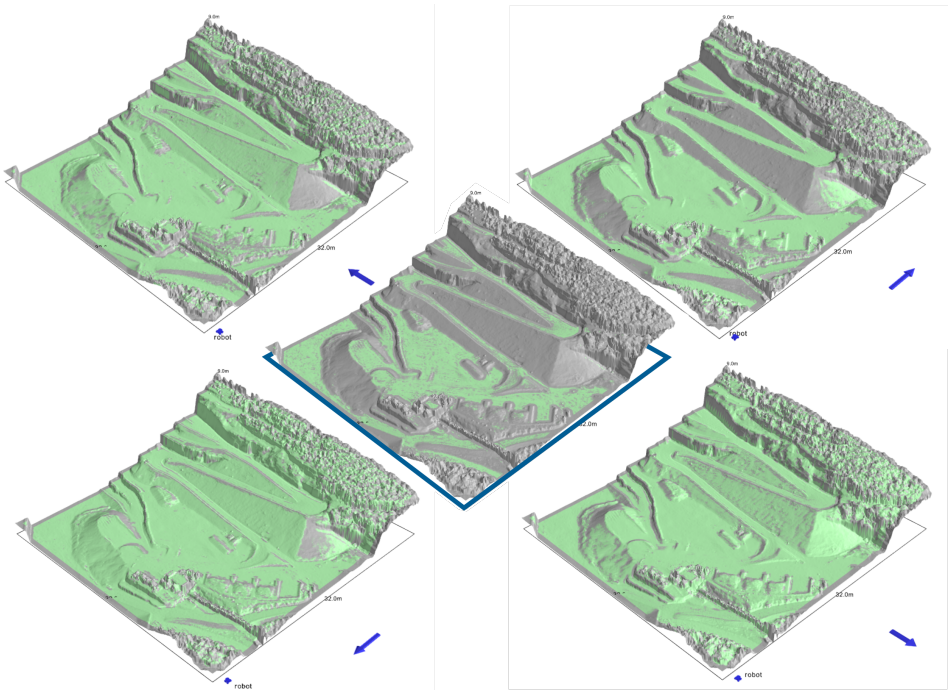

Figure 6: Traversability estimation for the mining quarry elevation map, for four orientations: $180°$ (left), $90°$ (up), $270°$ (down), $0°$ (right). Minimum traversability (blue frame) overlay over 32 possible orientations. For all figures, green indicates traversability and gray indicates non-traversability.

composed of segments $(e_1, \ldots, e_n)$ as $p((e_1, \ldots, e_n)) = \prod_{i=1}^{n} p(e_i)$, where $p(e_i)$ is the probability to traverse a segment.

We define a directed graph (N, E) where: nodes in N are placed on a regular square grid of side length 18 cm, and; edges in E connect each node with its 8 nearest neighbors. For each edge $e \in E$ we compute length and traversal probability $p(e)$. $p(e)$ is the output of the traversability classifier applied to a patch centered at the edge's origin and oriented along the edge direction (see Figure 7-right).

Figure 7-left illustrates the solution of the multi-objective problem of computing the bests paths with respect to length (to be minimized) and traversability (to be maximized). A rational agent would choose among the set of Pareto-optimal paths. In this context, a path is Pareto-optimal if there exist no alternative path that is both more traversable and shorter. The shortest path (represented in red in the following) is Pareto-optimal, but will have in most cases a very low traversability probability. The path with the maximum traversability (represented in green) will also be Pareto-optimal, but may be unnecessarily long; between the two extremes, a potentially very large set of Pareto-optimal paths exist, spanning the trade-off between traversability and length.

Real robot experiment We test how our approach applies to a real Pioneer 3-AT robot in an environment where a pair of bars (1 m length $\times$ 6 cm height) of width 6 cm and 8 cm respectively lie horizontally aligned on a flat floor with a small gap of 15cm in-between (see Fig. 8); the bars are fixed to the floor. We use an hand-generated heightmap representing the known geometry of the scene. The robot is tasked to move from the area below the leftmost bar to the area above it. There is no space to pass left of the bars, but plenty of space to pass on the right of the map. The robot is constrained to move along vertical paths over (or beside) the obstacles. To verify whether a given path is traversable, we teleoperate the robot following such path.We test three different trajectories: the most direct one where the robot will always get stuck (red); the one when it may pass in the space between the two bars by having wheels on one side climbing the bars, whereas the wheels on the other side passing in the gap, this path has a probability of about 50% that was empirically verified through several repetitions; and the longer but safest one going around the obstacles (green).

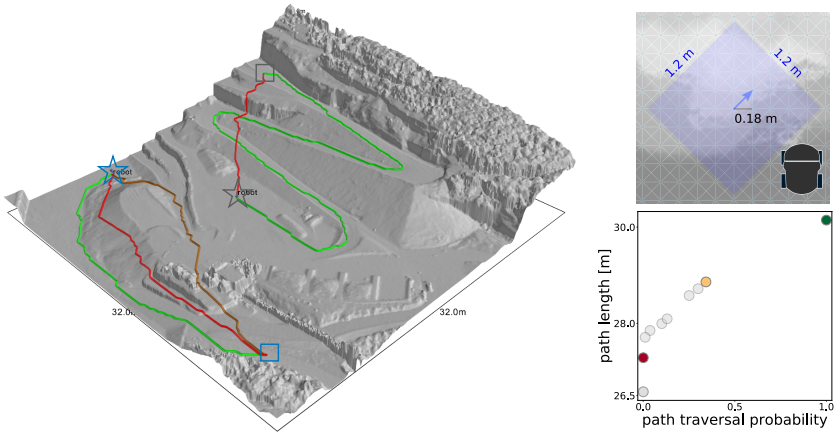

Figure 7: Left: a selection of Pareto optimal paths on the quarry map for two pairs of source (star) and target (square) locations. The paths are colored by traversal probability from red (non-traversable) to green (surely traversable). Right top: a portion of the planning graph on the quarry map. Nodes are placed on a regular grid of 18 cm. The blue edge's traversal probability is estimated by applying the classifier on a 1.2 m x 1.2 m patch (light blue) centered at the edge origin and directed along the edge. Robot's silhouette is shown for size comparison. Right bottom: The trade-off between path length and traversability for Pareto optimal paths between blue star and blue square. Colored dots correspond to paths drawn on the left figure.

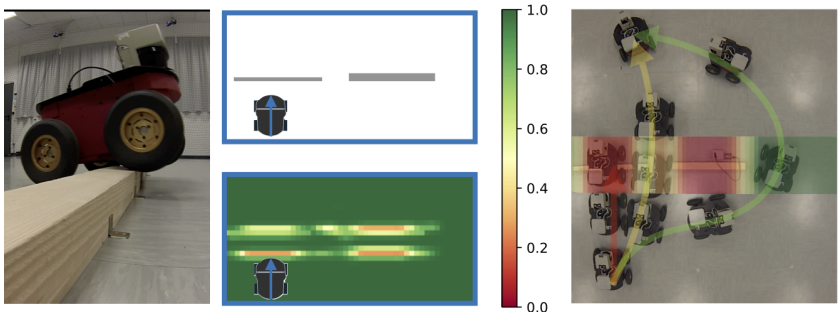

Figure 8: Left: Pioneer 3AT tries to pass over the bars. Center: height-map with two bars of width 6 cm and 8 cm (top) and patch traversability estimation moving upwards (bottom). Right: robot tries alternative routes. For all paths going straight upwards, the estimated traversal probability is displayed on top of the bars.

5 Conclusions and Perspectives

We presented a complete framework for traversability estimation that casts the problem as an elevation-map classification task. Classifiers trained on simulation data capture complex characteristics of different robot models and quickly estimate traversability maps on large unseen terrains.

We assumed that the terrain's 3D shape is the only factor influencing traversability; other factors may also be considered, such as compactness, friction, and instability. Our framework could handle these factors provided they can be simulated: for example, if terrains whose 3D shape suggests a loose gravel surface were simulated with lower friction, the classifier would automatically learn that steep slopes can't be negotiated on such surfaces; in this perspective, one may add additional inputs to the classifier (such as the visual texture) which may help differentiate the actual surface.

Our current approach does not capture the robot dynamics, such as the speed with which the robot approaches an obstacle: we limit our attention to slow robots operating on rugged terrains, where dynamic aspects have negligible impact. We consider the addition of such information as we move towards scenarios where complex locomotion is needed.

References

[1] R. Hadsell, M. Scoffier, U. Muller, Y. LeCun, and P. Sermanet. Mapping and planning under uncertainty in mobile robots with long-range perception. In *In Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1–6, 2008.

[2] U. E. and S. E. Traversability: A case study for learning and perceiving affordances in robots. *Adaptive Behavior*, 18:258–284, 2010.

[3] V. Molino, R. Madhavan, E. Messina, A. Downs, S. Balakirsky, and A. Jacoff. Traversability metrics for rough terrain applied to repeatable test methods. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1787–1794, Oct 2007. doi:10.1109/IROS.2007.4399438.

[4] C. A. Brooks and K. D. Iagnemma. Self-supervised classification for planetary rover terrain sensing. In *2007 IEEE Aerospace Conference*, pages 1–9, March 2007. doi:10.1109/AERO.2007.352693.

[5] M. A. Bekhti, Y. Kobayashi, and K. Matsumura. Terrain traversability analysis using multi-sensor data correlation by a mobile robot. In *2014 IEEE/SICE International Symposium on System Integration*, pages 615–620, Dec 2014. doi:10.1109/SII.2014.7028109.

[6] D. Silver, J. A. Bagnell, and A. Stentz. Learning from demonstration for autonomous navigation in complex unstructured terrain. *Int. J. Rob. Res.*, 29(12):1565–1592, Oct. 2010. ISSN 0278-3649. doi:10.1177/0278364910369715. URL http://dx.doi.org/10.1177/0278364910369715.

[7] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, J. Han, U. Muller, and Y. LeCun. Online learning for offroad robots: Spatial label propagation to learn long-range traversability. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007. doi:10.15607/RSS.2007.III.003.

[8] A. Chilian and H. Hirschmuller. *Stereo camera based navigation of mobile robots on rough terrain*, pages 4571–4576. 2009.

[9] M. Herbert, C. Caillas, E. Krotkov, I. S. Kweon, and T. Kanade. Terrain mapping for a roving planetary explorer. In *Proceedings, 1989 International Conference on Robotics and Automation*, pages 997–1002 vol.2, May 1989. doi:10.1109/ROBOT.1989.100111.

[10] H. Balta, G. D. Cubber, D. Doroftei, Y. Baudoin, and H. Sahli. Terrain traversability analysis for off-road robots using time-of-flight 3d sensing. In *In Proc. of the Int. Workshop on Robotics for Risky Environment - Extreme Robotics*, 2013.

[11] S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb, and R. Chatila. Autonomous rover navigation on unknown terrains functions and integration. In *Experimental Robotics VII*, ISER '00, pages 501–510, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-42104-1. URL http://dl.acm.org/citation.cfm?id=645627.757994.

[12] M. Schwarz and S. Behnke. Local navigation in rough terrain using omnidirectional height. In *ISR/Robotik 2014; 41st International Symposium on Robotics*, pages 1–6, June 2014.

[13] R. Hudjakov and M. Tamre. Aerial imagery terrain classification for long-range autonomous navigation. In *2009 International Symposium on Optomechatronic Technologies*, pages 88–91, Sept 2009. doi:10.1109/ISOT.2009.5326104.

[14] M. Längkvist, A. Kiselev, M. Alirezaie, and A. Loutfi. Classification and Segmentation of Satellite Orthoimagery Using Convolutional Neural Networks. *Remote Sensing*, 8(4), 2016.

[15] L. D. Jackel, E. Krotkov, M. Perschbacher, J. Pippine, and C. Sullivan. The darpa lagr program: Goals, challenges, methodology, and phase i results. *Journal of Field Robotics*, 23(11-12):945–973, 2006. ISSN 1556-4967. doi:10.1002/rob.20161. URL http://dx.doi.org/10.1002/rob.20161.

[16] M. Shneier, T. Chang, T. Hong, W. Shackleford, R. Bostelman, and J. S. Albus. Learning traversability models for autonomous mobile vehicles. *Autonomous Robots*, 24(1):69–86, 2008. ISSN 1573-7527. doi:10.1007/s10514-007-9063-6. URL http://dx.doi.org/10.1007/s10514-007-9063-6.

[17] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, pages 2149–2154 vol.3, Sept 2004.

[18] K. Perlin. Improving noise. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 681–682. ACM, 2002.

[19] M. Olano. Modified noise for evaluation on graphics hardware. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 105–110. ACM, 2005.

[20] A. Lagae, S. Lefebvre, R. Cook, T. DeRose, G. Drettakis, D. S. Ebert, J. P. Lewis, K. Perlin, and M. Zwicker. A survey of procedural noise functions. In *Computer Graphics Forum*, volume 29, pages 2579–2600. Wiley Online Library, 2010.

[21] R. M. Smelik, K. J. De Kraker, T. Tutenel, R. Bidarra, and S. A. Groenewegen. A survey of procedural methods for terrain modelling. In *Proceedings of the CASA Workshop on 3D Advanced Media In Gaming And Simulation (3AMIGAS)*, pages 25–34, 2009.

[22] SenseFly. Elevation datasets. https://www.sensefly.com/drones/example-datasets.html, 2016. URL https://www.sensefly.com/drones/example-datasets.html.

[23] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

[24] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. ISSN 0893-6080. doi:https://doi.org/10.1016/j.neunet.2014.09.003. URL http://www.sciencedirect.com/science/article/pii/S0893608014002135.

[25] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001. ISSN 1573-0565. doi:10.1023/A:1010933404324. URL http://dx.doi.org/10.1023/A:1010933404324.

[26] F. Chollet et al. Keras. https://github.com/fchollet/keras, 2015.

[27] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org.