Navigation Planning for Legged Robots in Challenging Terrain

Navigation Planning for Legged Robots in Challenging Terrain

Martin Wermelinger[1], Péter Fankhauser[1], Remo Diethelm[2], Philipp Krüsi[2], Roland Siegwart[2], Marco Hutter[1]

***Abstract*— This paper presents a framework for planning safe and efficient paths for a legged robot in rough and unstructured terrain. The proposed approach allows to exploit the distinctive obstacle negotiation capabilities of legged robots, while keeping the complexity low enough to enable planning over considerable distances in short time. We compute typical terrain characteristics such as slope, roughness, and steps to build a traversability map. This map is used to assess the costs of individual robot footprints as a function of the robot-specific obstacle negotiating capabilities for steps, gaps and stairs. Our sampling-based planner employs the *RRT** algorithm to optimize path length and safety. The planning framework has a hierarchical architecture to frequently replan the path during execution as new terrain is perceived with onboard sensors. Furthermore, a cascaded planning structure makes use of different levels of simplification to allow for fast search in simple environments, while retaining the ability to find complex solutions, such as paths through narrow passages. The proposed navigation planning framework is integrated on the quadrupedal robot StarlETH and extensively tested in simulation as well as on the real platform.**

I. Introduction

Legged robots show their full potential in rough and unstructured terrain, where they are superior to wheeled and tracked platforms. Their primary advantage is that they are able to overcome large obstacles compared to their body size. Additionally, they are not limited to follow smooth paths as they can walk sideways. Autonomous navigation in rough terrain offers a wide range of possibilities and applications, but is still a challenging task and a topic of ongoing research.

Safe and efficient navigation requires knowledge about the shape and the traversability of the environment. Estimating traversability in rough, nonplanar terrain is not a straightforward task. The traversability of the terrain can continuously vary between fully traversable and hardly traversable or not traversable at all. Furthermore, it is not only a property of the terrain, but it depends on the capabilities of the robotic platform.

This paper contributes a novel framework for traversability estimation and path planning specifically targeted at *legged* robot navigation in challenging terrain. Perceived terrain characteristics are analyzed with respect to the specific capabilities of a legged robot to estimate the traversability of its footprint. Additionally, we present a planner structure that enables fast but powerful planning by applying different levels of simplification.

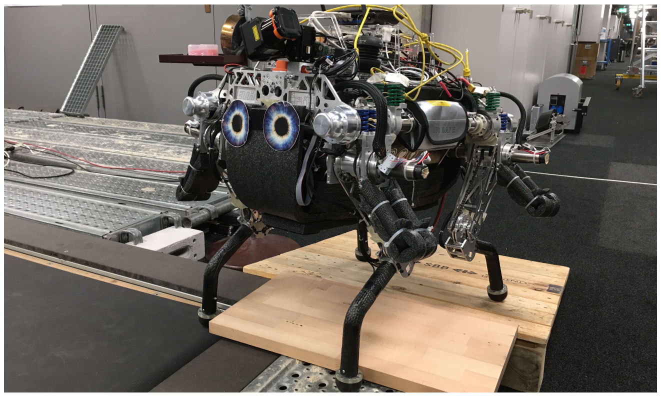

Fig. 1. The quadruped *StarlETH*, equipped with a rotating laser range sensor, during the navigation experiment described in Section V.

We estimate the traversability of the terrain on the basis of a discrete 2.5D *elevation map*, which we build using onboard sensing. The terrain characteristics *slope*, *roughness*, and *steps* are extracted by applying different filters on the elevation map. The results are combined into a *traversability map*. For planning, we use the sampling-based *RRT** algorithm [1], and evaluate the footprint of the robot for traversability at the relevant locations using the aforementioned map. A hierarchical planner architecture ensures adaption of the solution path during its execution, as new environment data is perceived. The planning process follows a cascaded structure that enables to find simple solutions quickly, while retaining the ability to solve difficult navigation problems.

The remainder of this paper is organized as follows. Section II compares the proposed framework with related work. Sections III and IV describe our approach to traversability estimation and the proposed planner architecture, respectively. In Section V, we experimentally demonstrate the suitability of the framework for rough terrain navigation. In section VI, we draw a conclusion and discuss future work.

II. Related Work

Chilian and Hirschmüller [2] presented rough terrain navigation for a legged and a wheeled robot applying a traversability estimation approach similar to ours, using stereo camera images to create the elevation map and planning with the *D* Lite* algorithm. To each cell of the elevation map, a traversability value corresponding to the surrounding terrain difficulty is assigned. However, they adapt the area for traversability estimation such that the robot can be approximated as a point and its orientation is neglected.

This work was supported in part by the Swiss National Science Foundation (SNF) through project 200021_149427 / 1, the National Centre of Competence in Research Robotics, and the armasuisse S+T UGV research program.

[1]Robotics Systems Lab (RSL), ETH Zurich, Switzerland, martiwer@mavt.ethz.ch

[2]Autonomous Systems Lab (ASL), ETH Zurich, Switzerland

In contrast to this, we use a smaller diameter to represent the traversability of a map cell, but additionally consider the cumulative traversability of the actual footprint of the robot. This allows to correctly deal with obstacles whose traversability is direction-dependent, and it enables planning through narrow passages.

Chestnutt [3] defined the planning problem for a legged robot as finding a discrete sequence of actions between the initial and the goal state. The choice of the type of actions determines the difficulty of the planning problem, but influences also how well the robot's capabilities are used. Whereas Chestnutt proposed individual stances as planning actions, we use torso poses and connect them using a locomotion controller during execution. The torso-based state space is only an approximation of the actual motion of the robot, but it reduces the planning complexity and enables fast planning over longer distances.

[4], [5], [6], and [7] divide path planning into separate tasks for footstep and torso motion planning to generate a sophisticated path over challenging terrain. However, they use external systems for terrain recognition and localization, and off-line computation for traversability assessment. Winkler et al. [8] extended these planning and control architectures by incorporating onboard perception, localization and computing equipment to re-plan actions and footholds online. Similar to their approach, we solely rely on onboard sensors and computation, resulting in a fully autonomous system. Instead of considering the traversability for the foothold only, we take the whole robot footprint traversability into account, which allows us to plan over longer distances without needing to specify the individual footholds.

The legged robot *BigDog* has been shown to be capable of walking autonomously in rough outdoor terrain [9]. A laser range sensor detects obstacles and generates a 2D costmap, which is used for path planning. Though the BigDog platform has a high capability to cross rough terrain, this planning framework is designed for generally flat terrain populated with large obstacles like trees and boulders. In contrast, our approach is designed to consider the actual traversability instead of simply finding a collision free path in a 2D environment.

III. TRAVERSABILITY ESTIMATION

The traversability of a specific terrain patch is dependent on the properties of the terrain itself (roughness, surface condition, etc.) and on the capabilities of the robot. To gather information about the environment, we use an exteroceptive sensor (such as a laser scanner, a Kinect, or a stereo camera) and build a robot-centric elevation map of the terrain [10]. The elevation map consists of a two-dimensional regular grid, where each cell stores a height value and variance. This map is continuously updated as the robot moves and perceives new data. Interpreting the elevation map for traversability is done at two different levels. First, a traversability map is created that assigns a local traversability value to each cell of the elevation map. On a second level, the traversability map is used to interpret the traversability of a certain robot pose by analyzing the corresponding footprint. These two levels are explained in the following sections.

A. Traversability Map

The traversability map uses the same regular grid as the elevation map. It locally describes the traversability for navigation, based on three different terrain characteristics: the local slope s, the local terrain roughness r, and the local step height h. We compute the values s, r, and h for each cell of the grid similar to [2], by using a circular region around the cell. The difference to their approach is that we do not assess these values for a circular region corresponding to the maximum diameter of the robot, but we let the radius of the circular region be as small as reasonably possible. For example, a smaller circular region has the advantage that we have a more precise representation of the terrain, which is essential to mark features like negotiable steps, small gaps and narrow corridors as traversable. The local traversability $t \in [0,1]$ combines the values computed for slope, roughness and steps of a cell as follows:

$$t = 1 - w_1 \frac{s}{s_{\text{crit}}} - w_2 \frac{r}{r_{\text{crit}}} - w_3 \frac{h}{h_{\text{crit}}}, \qquad (1)$$

where w_1, w_2 and w_3 are the weights which sum up to 1. The values $s_{\text{crit}}, r_{\text{crit}}$ and h_{crit} are the robot-specific maximum allowed values. If one of the terrain characteristics s, r or h exceeds its critical value, the traversability is set to $t = 0$.

B. Footprint Traversability

For path planning, we are interested in the traversability of particular robot footprints. As footprint, we use alternatively two different representations: either a generic rectangle that respects the robot's orientation, or a simplified, rotation-invariant circular footprint. The *footprint traversability* $t_f \in [0,1]$ is given by the mean of the local traversability values t of all cells within the footprint of the corresponding pose, where $t_f = 1$ indicates fully traversable and $t_f = 0$ means not traversable.

Beside the footprint traversability, we also consider hard constraints to decide if a footprint is traversable at all. These constraints are designed to include the ability of a legged robot to negotiate certain obstacles:

1) Steps and Gaps: Whereas high obstacles and tall steps should result in a non-traversable footprint, small steps and gaps (compared to the footprint) up to a certain maximum allowed width w_g can be overstepped by a legged robot and should yield a traversable footprint (see Fig. 2). To distinguish between traversable gaps and non-traversable tall steps we use the step value h. At each cell of the traversability map, h indicates if a step higher than the threshold locally exists. In general, both bottom and top cells of a step exhibit a high step value. For identifying traversable gaps, we only analyze top cells. We compute the height of all cells on a line with length w_g, perpendicular to the step direction, starting from the top cell. If the height difference between a cell on the line and the top cell drops below the threshold h_{crit}, we consider the step as gap and mark

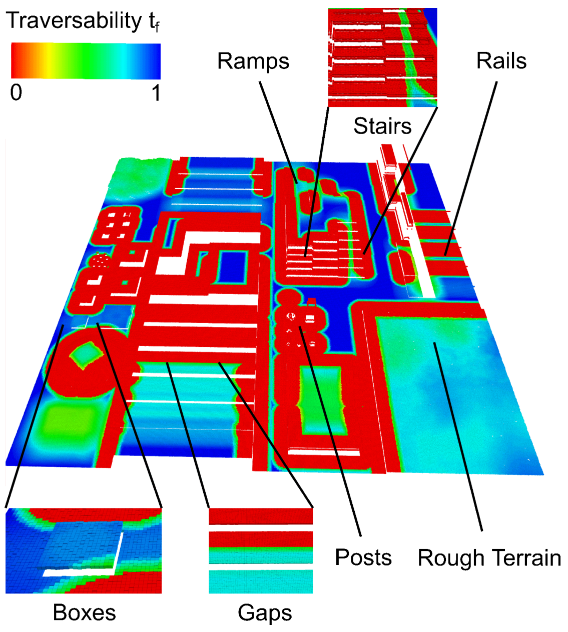

Fig. 2. The sample terrain used for parameter tuning consists of many different kinds of obstacles and features. For each cell of the grid the footprint traversability t_f of a circular footprint centered at this position is computed and represented with the following coloring: blue areas indicate full traversability, intermediate values are continuously decreasing from turquoise over green to yellow, and red areas are not traversable at all.

it as traversable. All steps within the footprint have to be traversable, otherwise it is considered as not traversable.

2) Terrain Inclination: To prevent the robot from slipping or tipping over, the maximum allowed overall terrain inclination within the footprint is limited. The inclination is computed analogously to the slope value s, but for a (larger) local area that corresponds to the footprint size. The footprint is considered as non-traversable if the inclination exceeds the maximum allowed slope. For example, a stair can have a too high overall inclination, although each step separately is traversable (see Fig. 2).

3) Local Roughness and Slope: A legged robot can overcome rough and steep areas that are spatially limited, e. g. the edge of a box (see Fig. 2). But steep or rough areas larger than the reachable step size w_g should be avoided and result in an untraversable footprint. Steep or rough areas are given by adjacent cells with a local slope or roughness value greater than the respective maximum allowed. If such an area lies within the footprint and has an extent larger than w_g, the footprint is considered as non-traversable.

C. Parameters

The traversability estimation depends on several parameters which are either directly derived by sensor and robot properties (footprint size, $s_{\text{crit}}, r_{\text{crit}}$, h_{crit}, etc.) or can be adjusted by the user (filter weights etc.). This allows configuring the traversability estimation for different purposes, either being more conservative or taking more risk, and it makes the approach adaptable to different robots. To facilitate the adjustment of the tunable parameters, we have created an artificial sample terrain composed of diverse obstacles in a surveyable area. With this sample terrain, we adjust the parameters of the traversability filters such that the ability of the robot to negotiate different obstacles is correctly represented. The results of parameter adaption for a quadruped robot are shown in Fig. 2.

IV. Path Planning

The purpose of the path planner is to find a short and feasible path between the start and the goal pose. We present two different simplification stages of the planning problem, which are combined to a hierarchical planner structure. This allows fast computation of simple solutions yet gives the planner the ability to find more complex solutions through narrow passages. We apply the sampling-based *RRT** algorithm [1] for efficient and fast planning over large distances. We use the state-of-the-art implementation provided by the Open Motion Planning Library (OMPL) [11] and adapt the collision checking of this algorithm by integrating our traversability assessment method. Additionally, we use tools provided by OMPL to smooth the solution path while retaining its validity.

A. Footprint Approximation

Different approximations of the robot footprint are employed within the hierarchical structure of the planner. Our most accurate representation of the footprint is a *rectangle* whose pose is given by a position (x, y) and a heading angle (ψ). This incorporates the legged robot's specific ability to move also sideways and rotate on the spot, and enables planning and moving even in complex situations and through narrow passages. The rectangular footprint is used for planning if the following approximations fail to generate a valid path.

Representing the footprint with a *circle* facilitates caching the computed footprint traversability values in a separate grid map. This accelerates the convergence of the path to the optimal solution and is particularly advantageous if the path is replanned on the same or partially same map. The robot's heading can be computed in a post-processing step, such that it is tangential to the path. A good first approach to approximating the actual footprint is taking the *circumscribed* circle, which represents the maximum diameter of the footprint. This guarantees the validity of the solution path, but may yield overly conservative solutions: narrow passages, which are traversable only with a specific orientation of the robot, will be labeled untraversable. To overcome this problem, we can use in a second step the *inscribed* circle of the rectangular footprint as approximation. Because this footprint is smaller than the actual one, validity is not guaranteed and has to be verified afterwards. This method has shown to be a good approach to quickly find a solution path through locally narrow passages. In more complex terrain, however, only planning with the rectangular footprint results in valid paths.

For reasons of robustness, it is desirable to avoid walking too close to obstacles if not necessary. We developed a method to increase the traversability cost of poses close to obstacles when the circumscribed circle with radius r is used to approximate the footprint. The radius of the circle is continuously inflated until the footprint is no longer valid or the inflation radius r_i has reached an upper bound $r_{\max}$. The footprint traversability t_f is corrected with a factor f_c:

$$f_c = \begin{cases} \dfrac{\dfrac{r_i - r}{r_{\max} - r} + n}{n+1} & \forall \ r_i < r_{\max} \\ 1 & \forall \ r_i \geq r_{\max} \end{cases}, \tag{2}$$

where $n > 0$ is a parameter to adjust the influence of the correction factor.

B. Cost Function

As a footprint of the robot can be represented with (x, y, ψ), the planning state space consists of three dimensions. In the case of a circular footprint, we can further reduce the planning state space by one dimension, since the footprint is invariant to the heading.

The aim of the path planner to find a short and smooth path with a high traversability is reflected in the cost function. The cost c of a path segment between the state $n = (x, y, \psi)$ and its neighbor $n' = (x', y', \psi')$ consists of the segment traversability cost c_{trav}, and the turning cost c_{turn}:

$$c = c_{\text{trav}} + c_{\text{turn}}. \tag{3}$$

To compute the different parts of the cost function, we need the segment properties: the length l, the traversability of the segment $t_{f_{\text{segment}}}$, and its moving direction ψ_{move}. The length l is given by Euclidean distance between n and n', and the traversability of the segment $t_{f_{\text{segment}}}$ is computed by using a footprint that is built by the convex hull of n and n'. The segment traversability cost is given by

$$c_{\text{trav}} = w_{\text{trav}} l \left(1 + w \left(\frac{1}{t_{f_{\text{segment}}}}\right)^p\right), \tag{4}$$

where w_{trav} is the respective weight and w and p are parameters to adjust the impact of the traversability compared to the segment length l. The turning cost is designed to explicitly punish walking sideways and backwards. Thus it consists of two parts, the cost for walking sideways $c_{\text{turn,s}}$ and walking backwards $c_{\text{turn,b}}$. To compute the turning cost we need the angle ψ_{move} of the moving direction between n and n':

$$\psi_{\text{move}} = \operatorname{atan2}(y' - y, x' - x). \tag{5}$$

Using the smaller difference angle $\Delta\psi$ between a state and the moving direction ψ_{move}, the turning costs are computed as

$$c_{\text{turn,s}} = -\left(\left(\Delta\psi - \frac{\pi}{2}\right)^2 + \left(\Delta\psi' - \frac{\pi}{2}\right)^2\right) \tag{6}$$

and

$$c_{\text{turn,b}} = -\left((\Delta\psi - \pi)^2 + (\Delta\psi' - \pi)^2\right). \tag{7}$$

The cost for walking sideways and backwards result in the turning cost

$$c_{\text{turn}} = w_{\text{turn,s}}\, c_{\text{turn,s}} + w_{\text{turn,b}}\, c_{\text{turn,b}} \tag{8}$$

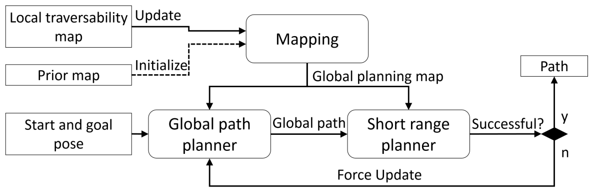

Fig. 3. Architecture of the path planning algorithm with the three main components mapping, global path planner, and short range planner.

with the weights $w_{\text{turn,s}}$ and $w_{\text{turn,s}}$. Since the turning cost is negative, it is only admissible to planning algorithms with search graphs without loops, like *RRT**. If the circular footprint is applied, the turning cost part c_{turn} of the cost function is omitted.

C. Hierarchical Structure

We have developed a hierarchical structure for the path planner that includes the tasks of updating the global planning map, generating a global path, and adapting the local path during execution. For efficient planning, we combine the advantages of rectangular and circular footprints by gradually increasing the accuracy of the approximation when necessary. Additionally, the path planner dynamically adjusts the planning bounds to reduce the planning time. The three main components of the planner are shown in Fig. 3 and described in the following sections.

1) Mapping: The traversability map is continuously computed in a local window around the current robot position. This local map is used to build and update the global planning map. A cell of the global map is updated if no prior traversability value is available or the elevation difference to the local map exceeds a threshold. The global planning map can be initialized with prior assumed traversability values which are updated as soon as actual perceived information is available. Unknown areas, which are not initialized, are assigned a low traversability value. This enables planning through unknown areas of the map but preferring already observed regions. Planning is performed directly on the global planning map. To retain consistency of the path, updating of the global planning map is suspended during planning.

2) Global Path Planner: The purpose of the global path planner is to find a complete global solution path from the start state (current robot pose) to the desired goal state using the latest update of the global planning map. The global path planner has a cascaded structure that applies different simplification levels successively until a solution is found (Fig. 4). First, the bounds for planning are set to contain the rectangular area between the start and goal state plus a certain margin. The path is planned using the circumscribed circle of the footprint. If planning is successful, the robot's heading is interpolated tangential to the path to generate the global path. Otherwise, we try to plan a path using the inscribed circle of the footprint. Again, if a solution path is found, the robot's heading is interpolated. Additionally, the validity of the solution path is verified using the rectangular

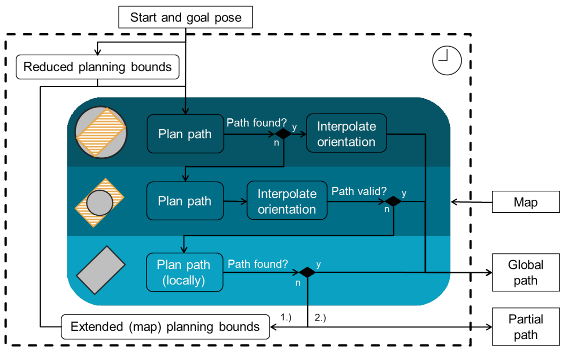

Fig. 4. The global path planner employs different footprint representations in a cascaded structure to find a global path between the start and goal pose.

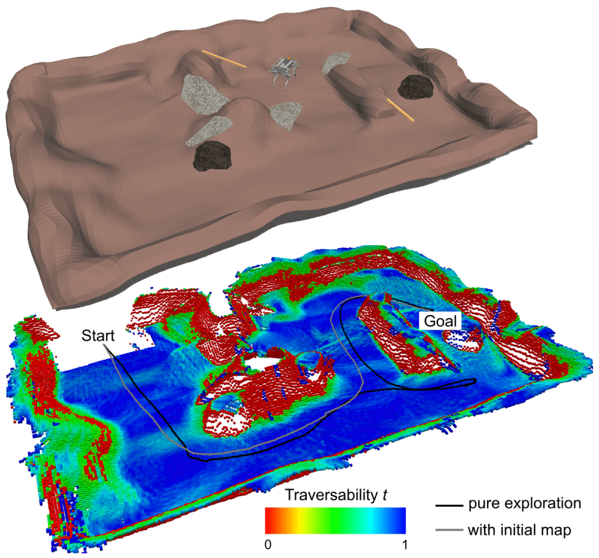

Fig. 5. An irregular terrain (top) consisting of banks, piles, rocks and blanks is used to test the planning framework for robustness and applicability in challenging terrain. The results of two simulated navigation experiments are shown together with the perceived elevation map colored according to the local traversability (bottom).

footprint. If parts of the path are found to be invalid, these are finally replanned with the rectangular footprint. Should this cascade not result in valid solution, the planning bounds are extended to the entire global planning map and the cascaded planning procedure is repeated. If no solution is found after the second run through the cascade, a partial path, leading to the closest valid position sampled around the goal state during planning, is used as replacement of the global path. The global path is not only computed once at the beginning of the planning procedure, but it is updated with a certain rate during execution.

3) Short Range Planner: Based on the global path, the short range planner produces a valid local path to be executed by the robot. It operates at a higher frequency than the global planner and uses the most up-to-date global planning map to verify the next segment of the global path starting from the current robot position. If the segment is valid, it is directly used for execution. Otherwise, the invalid segment of the global path is replanned. This is done with the same cascaded structure as described for the global path planner, but with locally restricted planning bounds. If a new valid path segment is found, it replaces the corresponding part of the global path. Should it not be possible to replan the invalid path segment locally, the short range planner forces an update of the global path.

V. Results

The traversability estimation and path planning algorithms were integrated on the quadruped StarlETH [12], which is designed for versatile, dynamic, and efficient motion. The robot is equipped with a Hokuyo UTM-30LX laser range sensor to perceive the environment (see Fig. 1). Localization [13] and state estimation [14] are performed online, resulting in an autonomous system. To implement the elevation mapping [10] and the traversability estimation, a universal grid map library was used [15]. The computation of the different terrain characteristics slope, roughness, and steps was implemented using a filter plugin-in representation, which allows to switch them on and off according to the actual demand. These filters are made for simple (re)configuration by setting the decisive parameters. To execute the planned path, the robot uses a statically stable walking gait, which enables crossing obstacles of the size of a robot foot. For negotiating larger obstacles like steps and gaps, an additional foothold planner would have to be incorporated, which is not part of this work.

A. Simulation

The complete traversability assessment and planning framework was tested with various terrains and obstacles in simulation. Fig. 5 (top) shows an irregular terrain composed of rocks, blanks, walls, piles and uneven floor, covering an area of 7x10 m. In this scenario, the task was to navigate autonomously to a defined goal pose. Fig. 5 (bottom) shows the global map built during navigation and the traveled paths planned with (gray) and without (black) using an initial map. In pure exploration mode, the global path outside the field-of-view heads directly to the goal position since this is the shortest connection. This led to exploring dead ends, but by replanning the global path when no local path was found (or at least every 60 s) the robot finally succeeded to reach the desired goal pose. The local path was verified and, if necessary, updated every 10 s. The robot needed 512.3 s to plan and execute the complete path, which had a final length of approx. 17.8 m. Providing the planner with a rudimentary initial map containing the outline of the walls and piles resulted in a more direct path without exploring dead ends. The path had a length of 13.4 m and was traveled in 363.8 s. The initial values of the global map are updated with actual measurements when the respective areas appear in the field-of-view. Hence, the system can handle inaccuracies of the initial map, like missing obstacles or shortcuts.

B. Real Platform

The navigation planner has also been experimentally validated on the real platform. We created an environment with

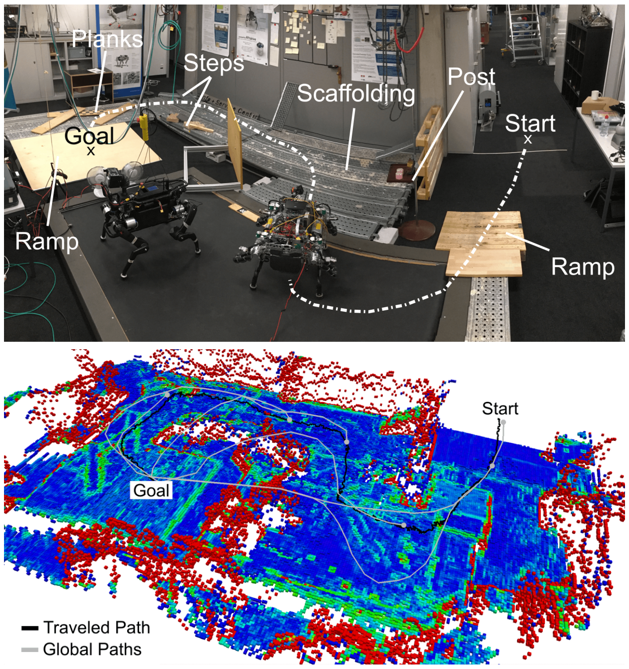

Fig. 6. Irregular terrain used for trial (top) and resulting global planning map (bottom). Shown is the finally traveled path from the start to the goal position (black) and the updated global paths computed during navigation (gray). Each start position for global path replanning is indicated with a dot.

ramps and scaffolding, cluttered with larger size obstacle that have to be bypassed. Additionally, obstacles that can be negotiated by the robot, like steps and planks, were spread on the ground. Fig. 6 shows the global planning map acquired during the experiment. The robot successfully reached the goal, traveling a total distance of 12.7 m. The entire navigation including planning and execution took 351 s. Since no prior knowledge had been provided to the planner, the global path significantly changed every time it was updated. However, the short range planner ensured the local validity of the path. Overall, this experiment showed that the traversability assessment and path planning framework can handle the noise of the on-board sensing and localization system while guiding a legged robot through challenging terrain.

VI. Conclusion

In this paper, we presented a framework for traversability estimation and path planning that is especially suited for legged robot navigation. The traversability is computed by applying filters to a discrete elevation map to extract three characteristics of the terrain: slope, roughness, and steps. For path planning, the traversability values beneath a robot footprint are evaluated, incorporating the specific ability of a legged robot to overcome obstacles such as steps, gaps, and locally steep and rough terrain. We developed a navigation architecture that allows adapting the executed path to newly perceived obstacles or shortcuts. Navigation can be performed in a pure exploration mode, but we can also initialize the traversability estimation with prior information. The planner itself takes advantage of a cascaded structure with different levels of simplification. Thereby, it remains fast in finding simple solutions, but it can also find complex paths through narrow passages. The proposed approaches have shown their suitability for autonomous navigation in simulation as well as in real experiments with the quadruped robot StarlETH.

For further improvement of rough terrain navigation, the path planner could choose different gaits for each path segment. Depending on the estimated traversability, the gait could be switched between planning each footstep separately, applying a static walking gait, or using a more dynamic trotting gait. This would allow to keep the efficient planning method and to increase the complexity only if necessary.

References

[1] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[2] A. Chilian and H. Hirschmüller, "Stereo camera based navigation of mobile robots on rough terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 4571–4576.

[3] J. Chestnutt, "Navigation Planning for Legged Robots," PhD Thesis, Carnegie Mellon University, Pittsburgh, PA, 2007.

[4] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, planning, and control for quadruped locomotion over challenging terrain," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 236–258, 2011.

[5] M. Zucker, N. Ratliff, M. Stolle, J. Chestnutt, J. A. Bagnell, C. G. Atkeson, and J. Kuffner, "Optimization and learning for rough terrain legged locomotion," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 175–191, 2011.

[6] J. Z. Kolter, M. P. Rodgers, and A. Y. Ng, "A control architecture for quadruped locomotion over rough terrain," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008, pp. 811–818.

[7] M. A. Arain, I. Havoutis, C. Semini, J. Buchli, and D. G. Caldwell, "A comparison of search-based planners for a legged robot," in *9th Workshop on Robot Motion and Control (RoMoCo)*, 2013, pp. 104–109.

[8] A. W. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 5148–5154.

[9] D. Wooden, M. Malchano, K. Blankespoor, A. Howardy, A. Rizzi, M. Raibert *et al.*, "Autonomous navigation for bigdog," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 4736–4741.

[10] P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Y. Siegwart, "Robot-centric elevation mapping with uncertainty estimates," in *International Conference on Climbing and Walking Robots (CLAWAR)*, 2014.

[11] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012.

[12] M. Hutter, C. Gehring, M. Bloesch, M. A. Hoepflinger, C. D. Remy, and R. Siegwart, "StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion," in *International Conference on Climbing and Walking Robots (CLAWAR)*, 2012.

[13] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP Variants on Real-World Data Sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[14] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. Hoepflinger, R. Siegwart, and Others, "State estimation for legged robots on unstable and slippery terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[15] P. Fankhauser and M. Hutter, "A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation," in *Robot Operation System (ROS): The Complete Reference (Volume 1)*, A. Koubaa, Ed. Springer, 2016, ch. 5.