Path Planning for UGVs Based on Traversability Hybrid A*

Marius Thoresen, *Student Member, IEEE*, Niels Hygum Nielsen, Kim Mathiassen, *Member, IEEE*, and Kristin Y. Pettersen, *Fellow, IEEE*

***Abstract*—In this letter, a new method of path planning for unmanned ground vehicles (UGVs) on terrain is developed. For UGVs moving on terrain, path traversability and collision avoidance are important factors. If traversability is not considered, the planned path may lead a UGV into areas that will cause rough vehicle motion or lead to the UGV getting stuck if the traversability is low. The proposed path planning method is based on the Hybrid A* algorithm and uses estimated terrain traversability to find the path that optimizes both traversability and distance for the UGV. The path planning method is demonstrated using simulated traversability maps and is compared to the original Hybrid A* algorithm. The method is also verified through real-time experiments in real terrain, further demonstrating the benefits of terrain traversability optimization using the proposed path planning method. In the experiments, the proposed method was successfully applied for autonomous driving over distances of up to 270 m in rough terrain. Compared with the existing Hybrid A* method, the proposed method produces more traversable paths.**

***Index Terms*—Motion and path planning, autonomous vehicle navigation, field robots.**

I. Introduction

ON ROUGH terrain, the environment can rarely be clearly segmented into obstacles and free space. Instead, for an unmanned ground vehicle (UGV), the terrain will have varying degrees of traversability depending on both the characteristics of the terrain itself and the terrain-traversing capabilities of the UGV [1], [2]. Autonomous driving on rough terrain is difficult because it is challenging to both correctly classify the terrain and quantify its traversability [3].

Manuscript received August 12, 2020; accepted December 19, 2020. Date of publication February 1, 2021; date of current version February 17, 2021. This letter was recommended for publication by Associate Editor Prof. Marco Morales and Editor Prof. Nancy Amato upon evaluation of the reviewers' comments. This work was supported in part by the Research Council of Norway through the Centres of Excellence funding scheme under Project 223254 - NTNU AMOS and in part by the Norwegian Defence Research Establishment. *(Corresponding author: Marius Thoresen.)*

Marius Thoresen and Kristin Y. Pettersen are with the Centre for Autonomous Marine Operations, and Systems (NTNU AMOS), Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim NO-7491, Norway, and also with the Norwegian Defence Research Establishment, Kjeller NO-2007, Norway (e-mail: marius.thoresen@ntnu.no; Kristin.Y.Pettersen@ntnu.no).

Niels Hygum Nielsen is with the Norwegian Defence Research Establishment, Kjeller NO-2007, Norway (e-mail: niels-hygum.nielsen@ffi.no).

Kim Mathiassen is with the Norwegian Defence Research Establishment, Kjeller NO-2007, Norway, and also with the Department of Technology Systems, University of Oslo, Kjeller NO-2007, Norway (e-mail: kimmat@ifi.uio.no).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3056028, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3056028

Traditional path planning algorithms rely on a world model consisting of *obstacles* and *free space*, and the planning objective typically involves finding paths that minimize the travel distance. Thus, directly using classical algorithms for path planning on rough terrain is challenging. For this reason, path planning on terrain is typically performed using world models that capture more information than only obstacle/free-space, such as 3D elevation maps, maps with precalculated traversability scores, segmented maps with terrain types or similar information, or combinations of these [1]. The methods for traversability mapping can be divided into two main categories, *geometry-based* and *appearance-based* [3]. Geometry-based methods use a 3D model of the surface for finding traversability and has been used for UGVs such as Mars rovers for several years [4]. Appearance-based methods instead use cameras to segment the terrain into categories such as road, grass and trees based on the appearance. There has been a large development in this field in the recent years based on the innovations in the field of Deep Learning [5]. In recent approaches, appearance- and geometry-based methods have also been combined to create joint traversability models [6]. With terrain traversability maps, parameters other than distance can be optimized. For example, optimizing the accumulated traversability is a strategy that can help strike a balance between ensuring maximum safety and avoiding excessive conservatism.

Path planning methods for driving on terrain can be divided into two main categories, namely, grid search planners and sampling-based planners [7]. Grid-based path planners can find optimal paths based on nonuniform cost maps and form complete planning methods, meaning that they will generally find a solution if one exists for the given cost map. They are, however, less suited for kinematically constrained vehicles because they lack mechanisms for handling (for example) minimum turning radii in cost maps.

A* is a widely used grid search algorithm that can address nonuniform cost fields [8]. However, the paths produced by this algorithm adhere strictly to a grid pattern and thus are not smooth. Field D* [9] is similar, but its paths are not restricted to a grid pattern, and thus, it produces smoother paths. Field D* has also been used for real UGVs on terrain [10]; a grid search with Field D* has been used as a decision variable for selecting a short-horizon feasible path. The fast marching method (FMM) [11] is another grid method which forms the basis of a class of any-angle path planners such as the FM^2 planner [12] that plans smooth paths in a grid. The method

works by calculating a one-to-all time-of-arrival estimate for each cell of the entire grid, and by sequentially differentiating this field a smooth path can be found from any cell to the source. The FMM has been used for nonuniform cost maps based on terrain traversability [13], [14] to find the cost optimal path. The FMM has also been extended for application to kinematically constrained vehicles, but only for binary obstacle maps; it has not been shown to work with cost maps [15], [16]. Another method closely related to the FMM are the artificial potential field (APF) methods, which also work by calculating a potential that attracts towards the goal and repulses from obstacles. The FMM itself calculates a minima-free attractive scalar potential. FM^2 adds a repulsive scalar potential, in order to avoid obstacles with a safety margin and to provide smooth paths. FM^2 is therefore an APF-like method, and the main difference is that FM^2 calculates a combined attractive and repulsive scalar potential which is then sequentially differentiated to find a path, while the APF methods separate the attractive and repulsive potentials and add the gradients of these two potentials instead [17]. In general, the APF methods suffer from the risk of getting stuck in local minima but there are techniques to avoid this [18]. An advantage of the APF methods is that they also can be used for collision avoidance with dynamic obstacles [19] which has not yet been shown for the FMM.

The other main class of path planning methods consists of sampling-based planners, which sample the search space and then combine the selected samples to form a solution. As the basis of the sampling process, sampling-based planners can use *motion primitives*, which are short, simple and feasible paths corresponding to single driving maneuvers. Sampling with motion primitives ensures that the solution will adhere to the kinematic constraints of the vehicle and will be feasible for driving. The Rapidly-exploring Random Trees (RRT) algorithm [20] uses random sampling and combines the resulting samples into a search tree. It is asymptotically complete, i.e., as the number of samples goes to infinity, the probability of finding a solution if one exists approaches 1. RRT* is an extension of RRT that finds the shortest feasible path and has been widely used recently, e.g., by [21], [22], where the latter used RRT* in an off-road scenario. Another sampling-based approach to terrain driving was presented in [23]. This approach samples a fixed set of motion primitives and selects the best one. The method uses a receding horizon strategy; the vehicle follows the current path for a while, and then it plans and switches to a new path without completing the previous path. However, because each path is a single motion primitive, the planning horizon is limited, and the method is susceptible to becoming stuck in a local minimum.

Grid search and sampling methods have been combined into various hybrid methods to achieve cost-optimal solutions while maintaining kinematic feasibility. For instance, in the Hybrid A* path planning algorithm [24] an A* grid search is performed first, and the results are then used to guide a continuous-state tree search that resembles the tree search of the RRT algorithm. In [25], it was postulated that Hybrid A* could be extended to consider cost based on a very simple traversability model, but that approach has not been evaluated on the basis of either simulations or real experiments. Recently, Hybrid A* was used in terrain driving for a small UGV with a high-fidelity vehicle model [26] by applying a receding horizon strategy. However, this method has a short look-ahead distance and neglects to provide a cost-to-go estimate when prioritizing nodes, which limits the planning horizon. A different type of hybrid approach was adopted in [27], where Field D* was used to find a rough path that was then directly refined to ensure kinematic feasibility at the expense of sacrificing optimality.

While the use of a traversability grid search as a basis for a sampling-based search that considers kinematic constraints has been attempted before, the previous approaches either lack a sufficiently long planning horizon [23], [26] or do not consider optimality in the sampling search [27], [28]. To address these problems, in this letter, we develop the Traversability Hybrid A* algorithm. This algorithm finds kinematically feasible and near-optimal paths in real time that are suitable for use by a UGV navigating rough terrain. The algorithm is based on the Hybrid A* algorithm, but we substitute the A* grid search heuristic for a fast marching grid search that explicitly considers a dynamic cost of the terrain. Then, we apply a tree search procedure as in the original Hybrid A* using motion primitives. However, instead of using the length of the motion primitives as the cost we use the accumulated traversability cost along the primitives in the tree search. With the combination of these two methods, the fast marching method provides an estimate of the weighted traversability cost-to-go, ensuring a long planning horizon, and the continuous-state tree search with motion primitives ensures the kinematic feasibility of the solution.

The remainder of this letter is organized as follows: Section II describes the development of the new Traversability Hybrid A* algorithm. In Section III, the new algorithm is evaluated and compared to the regular Hybrid A* algorithm using artificially generated traversability maps. In Section IV, we report the results of real-life experiments conducted to evaluate the new algorithm. This section addresses the use of the Traversability Hybrid A* algorithm for autonomous driving on terrain, and we also directly compare it to the original Hybrid A* algorithm. In Section V, we present the conclusion and future work.

II. Path Planning Method

In this section, the new Traversability Hybrid A* path planning method is presented. This algorithm plans near-optimal paths based on nonuniform cost maps derived from traversability estimates. The procedure for cost map optimization is an extension of the original Hybrid A* algorithm [24], which finds the shortest path from a start point to a goal point by minimizing the length of the path S:

$$Length = \int_S ds \tag{1}$$

In our approach, we operate with a cost map C with nonnegative cost values. The accumulated cost for a path S in this map can be found by integrating the cost values c along the path:

$$Cost = \int_S c\, ds \tag{2}$$

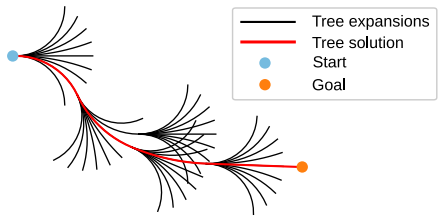

Fig. 1. The tree search process of Hybrid A*. Motion primitives are used to expand the tree from the start node, and the most promising nodes are used for further expansion.

Our objective is to find the path S that minimizes this accumulated cost. The cost map C is derived from a traversability map which we assume is given. This traversability map must be normalized so that it contains traversability values in the range of [0.0, 1.0] where 0.0 is the most traversable, increasing values up to 1.0 indicate lower traversability, and values exactly equal 1.0 indicate obstacles. The new Traversability Hybrid A* algorithm does not consider the creation of such a traversability map. However, in Section IV we present results from live experiments where we describe the specific method used for traversability estimation used in the experiments.

The cost map is created by a linear mapping of the traversability values to cost values. High traversability are assigned low cost and low traversability are assigned high cost so that our new method essentially finds the accumulated traversability. The amount of scaling in the mapping is tunable and decides how much high traversability will be prioritized. By selecting a high cost scaling, high traversability is prioritized at the expense of longer paths. By selecting a lower cost scaling, shorter distance is prioritized at the expense of lower traversability. By selecting no cost scaling at all, all traversability values are assigned the same cost, and our new algorithm essentially reduces to the original Hybrid A* algorithm that finds the shortest path. The exact mapping of traversability to cost is described in detail in Section II-C.

A. *Hybrid A* Path Planning Algorithm*

The regular Hybrid A* algorithm [24] which our method is based on, has three steps in its process. First, it uses the obstacle map to perform an A* shortest-path grid search from the goal position to all cells of the map. This is referred to as the grid heuristic and is used to give an estimate of the remaining distance from any point to the goal position. Second, it performs a continuous-state tree search from the start pose, or the start *node*. The nodes are expanded with kinematic motion primitives representing feasible driving maneuvers to create new nodes. These nodes are then evaluated by the sum of the driven distance, the *cost-so-far*, and the estimated remaining distance, the *cost-to-go*. The node with the lowest cost score is selected for a new expansion and the process is repeated until the goal is reached. As the search tree is composed of motion primitives, the path is guaranteed to be kinematically feasible to drive. An illustration of the tree search process with motion primitives can be seen in Fig. 1. Third, after a path is found, it is further refined through a local gradient-based optimization to make the path smoother and avoid unnatural swerves that can result from the tree search process. Together, these steps give the Hybrid A* algorithm the ability to find smooth, near-optimal paths that account for kinematic constraints.

Hybrid A* is chosen as a basis for our new method because it uses a grid heuristic, which enables the tree search to efficiently explore the search space towards the goal. Moreover, this grid search can be easily replaced with a more sophisticated grid search that also considers the traversability cost, effectively finding an estimate of (2) instead of (1). This modification can be made without adding much complexity to the algorithm or significantly increasing the processing time. Another reason for using Hybrid A* as a basis is that it runs in real time and was developed to work in unknown environments, which are both requirements for our application.

Our Traversability Hybrid A* algorithm is developed by adding three extensions to the original Hybrid A* algorithm. First, we use a fast marching grid search as a heuristic instead of the A* grid heuristic for reasons that will be detailed in Section II-B. Moreover, the new heuristic uses the traversability cost map to find a cost-to-go estimate based on (2) instead of finding the estimated distance to the goal. Second, during the tree search, the nodes are evaluated and prioritized in terms of the accumulated traversability cost and the new grid heuristic instead of being prioritized by distance. Third, a term that accounts for the traversability cost is added to the optimization objective function to ensure that smoothing is not performed at the expense of traversability.

B. *Fast Marching Grid Heuristic*

The grid heuristic of Hybrid A* provides an estimate of the cost-to-go from any point in the search space to the goal while avoiding obstacles. This grid heuristic allows Hybrid A* to prioritize the nodes with the lowest sums of the cost-to-go and the cost-so-far. If the heuristic is accurate, fewer nodes must be explored to obtain a solution, and the computation time is improved. However, because the A* search allows movement only between the centers of neighboring cells in the grid, the estimated distance is not a perfect reflection of the true distance between points in the map; it overestimates the true Euclidean distance between grid cells that are not immediate neighbors. The accuracy of the estimated distance also varies based on the direction between cells in the grid, causing the heuristic to favor certain directions.

The fast marching method is similar to the A* grid search, but it finds a numerical approximation of the Euclidean distance from a source cell to any other that avoids obstacles [11] instead of the grid distance that A* finds. Since the fast marching method approximates the Euclidean distance between any grid cells, the directional bias of A* is close to eliminated. Motivated by the improvements of the fast marching method over A*, we use the fast marching method as the heuristic in our new method.

C. *Fast Marching With Terrain Traversability*

Our goal with the new method is to find optimal paths in nonuniform cost fields; therefore, it is important for the grid

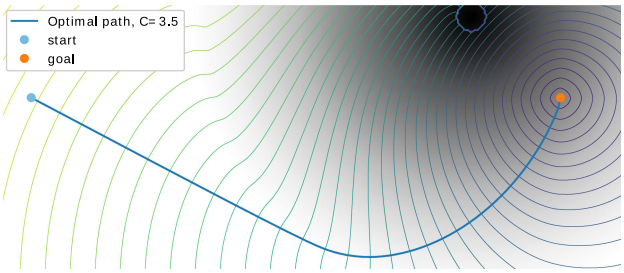

Fig. 2. Fast marching is applied to a grayscale cost map with a source point shown in orange. The level of gray indicates cost value, with white being 1.0 and black being 3.5. The contours show the cost of moving from the source to any point in the map starting with a cost of 0 at the source. Also shown is the path with optimal cost from a starting point to the source.

heuristic to also account for nonuniform costs. The A* grid heuristic is able to handle nonuniform costs so that it finds the cost of moving from the source to any cell instead of purely the distance. The fast marching method also handles nonuniform cost the same way, and is therefore capable of finding the cost of moving from a source to any cell, but without the directional bias of A*.

Fig. 2 illustrates this principle with nonuniform cost values. The level of gray represents the cost values of the map, where white indicates a nominal cost of 1.0, black indicates a cost of 3.5, and the various gray levels correspond to continuously varying costs between these two levels. A goal point is selected in the right part of the map, and fast marching is applied with this point as the source in order to obtain the cost of moving from the source to any point in the map. A contour plot of the fast marching cost from the goal point is plotted on top of the map. The fast marching cost starts at 0 represented by a dark blue color of the level sets, and green and yellow colors represent increasing cost. We see that the level sets become narrower as the cost increases, indicating that the fast marching cost increases more rapidly per distance here. Also, a blue path can be seen: this is the path that minimizes the accumulated cost in terms of (2), starting from the point to the left in the map and ending at the selected goal point. This path is obtained by starting at the starting point, following the negative gradient of the fast marching field and iterating in small steps towards the goal. Essentially, this process describes how the fast marching method can be used for path planning in a nonuniform cost field [12].

The only reason not to simply use the fast marching method directly for path planning is because it does not address vehicle kinematic constraints. Fig. 2 shows an optimal path from one point to another. For a vehicle to follow this path, it would need to be aligned with the optimal path at the starting point, but this will not always be the case; thus, the fast marching solution cannot necessarily be used directly. Also, the fast marching method can produce paths that have sharper turns than a given vehicle is able to follow.

To use the estimated traversability in conjunction with cost, the traversability values of the given traversability map must be mapped to cost values. We use a linear mapping from the normalized traversability to cost values. We define a mapping such that the most traversable areas with traversability values of 0.0 are mapped to a cost of 1.0 as the minimum cost, and the least traversable areas with values up to 1.0 are mapped to a parameter $C_{\max}$. The mapping of traversability value T for each cell can be expressed as

$$C = 1.0 + (C_{\max} - 1.0)\, T \tag{3}$$

After the linear mapping, the cells with traversability values of exactly 0.0 are interpreted as obstacles and are therefore set to infinity to ensure that these cells are avoided in the planning. The parameter $C_{\max}$ is a design parameter that can be selected in order to affect how traversability is prioritized in the planning. By increasing this, low traversability is penalized more in the planning and high traversable areas are prioritized more. By lowering $C_{\max}$ to 1.0, the cost of all non-obstacle cells are set to 1.0 and our Traversability Hybrid A* algorithm reduces to the regular Hybrid A* algorithm that finds the shortest path. The second term of (3) is therefore the main theoretical contribution of this letter.

D. Tree Search and Local Optimization

The tree search of Traversability Hybrid A* is conducted as in the original Hybrid A* algorithm, except that the cost-so-far is calculated using the traversability cost according to (2) instead of (1). In the original Hybrid A* algorithm, a local optimization is applied to the path in order to further smooth the path and reduce unnatural vehicle swerves that can arise due to rapid changes in curvature of the path. The smoothing is, however, not strictly necessary as the paths are already kinematically feasible and both Hybrid A* and THA* would still be viable without any smoothing. In our approach, we conduct a local optimization mostly the same way as in the original Hybrid A* algorithm. However, we have dropped two terms of the original objective function that deals with distance to obstacles, as this caused the optimization to not converge when using cost maps.

Given a path $\mathbf{x}_i = (x_i, y_i),\ i \in [1, N]$ and the quantities $\Delta\mathbf{x}_i = \mathbf{x}_i - \mathbf{x}_{i-1}$ and $\Delta\phi_i = |\arctan\frac{\Delta y_{i+1}}{\Delta x_{i+1}} - \arctan\frac{\Delta y_i}{\Delta x_i}|$, our objective function reduced from the objective function of Hybrid A* [24] is given by

$$O_{\mathrm{HA}*} = w_s \sum_{i=1}^{N} (\Delta\mathbf{x}_{i+1} - \Delta\mathbf{x}_i)^2 + w_\kappa \sum_{i=1}^{N} \left(\frac{\Delta\phi_i}{|\Delta\mathbf{x}_i|} - \kappa_{\max} \right)^2 \tag{4}$$

where w_s is a weight for penalizing non-smoothness of the path, w_κ is a weight that penalizes curvature of the path and thus enforces the kinematic constraints of the vehicle and $\kappa_{\max}$ is the maximum allowed curvature of the path.

In our optimization, we use the objective function of (4) but add a term to avoid that the smoothing is made at the expense of the traversability cost properties of the path. In other words, the smoothing should not be made by taking shortcuts through high cost areas. Given the cost map with cost values $c_{\mathrm{trav}}(\mathbf{x})$, we define this additional objective term as follows:

$$O_{\mathrm{trav}} = w_{\mathrm{trav}} \sum_{i=1}^{N} c_{\mathrm{trav}}(\mathbf{x}_i) \tag{5}$$

To use this term in the optimization process, we perform numerical differentiation of the cost map to obtain a gradient map, which we use in conjugate gradient descent optimization

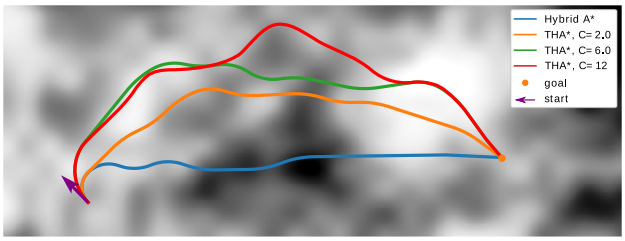

Fig. 3. Four simulated solutions on a Perlin traversability map. The purple arrow indicates the starting pose of the vehicle. The path lengths and traversability values are listed in Table I.

TABLE I
PATH LENGTH, ACCUMULATED TRAVERSABILITY AND AVERAGE TRAVERSABILITY FOR THE FOUR PATHS SHOWN IN FIG. 3

	Length	Acc. trav.	Avg. trav.
Hybrid A*	81.0 m	40.9	0.505
THA*, $C_{\max} = 2.0$	87.5 m	29.4	0.336
THA*, $C_{\max} = 6.0$	99.7 m	28.9	0.290
THA*, $C_{\max} = 12.0$	107.1 m	30.3	0.283

TABLE II
AVERAGE PATH LENGTH, ACCUMULATED TRAVERSABILITY, AVERAGE TRAVERSABILITY AND STANDARD DEVIATION FOR THE AVERAGE TRAVERSABILITY FOR 10 RANDOM SIMULATIONS

	Length	Acc. trav.	Avg. trav.	Std. dev.
Hybrid A*	107.3 m	52.6	0.491	0.082
THA*, $C = 2$	112.0 m	35.8	0.320	0.072
THA*, $C = 6$	129.0 m	35.2	0.273	0.058
THA*, $C = 12$	145.2 m	38.5	0.265	0.12

together with the $O_{\mathrm{HA}*}$ objective function of (4). Determining the values for the weight parameters w_κ, w_s and w_{trav} is best done empirically by finding a desired balance between the amount of smoothing determined by w_κ and w_s and the resistance to moving the path away from the low cost areas determined by w_{trav}. In practice, a starting point for tuning can be starting with a small smoothing weight w_s and large weights w_κ and w_{trav} which means the path is mostly unaffected by the optimization and then gradually increase w_s until desired smoothing is achieved. The result is a path that is smoother than the original, while it still maintains the cost properties.

III. SIMULATION

In this section, we present the application of the proposed Traversability Hybrid A* path planner to simulated traversability maps and compare the results to those of the original Hybrid A* planner to illustrate how different values of the cost parameter $C_{\max}$ affect the produced paths. The findings reveal that the new method plans kinematically feasible paths with higher total traversability.

In the simulation setup, the given normalized traversability maps contain traversability values between 0.0 and 1.0, where 1.0 indicates highest traversability and 0.0 indicates obstacles. Simulations were performed with simulated traversability maps generated using Perlin noise [29]. Perlin noise is continuous in two dimensions and can be generated with different frequency values to create noise at different scales. By combining noise maps of different scales, maps that mimic real terrain traversability can be created.

We applied the original Hybrid A* algorithm and the proposed Traversability Hybrid A* (THA*) algorithm to the same planning cases. Because Hybrid A* takes binary obstacle maps as input, the continuous traversability maps were thresholded such that only 0.0 values were interpreted as obstacles. It should also be noted that Hybrid A* can be considered a special case of THA* with $C_{\max} = 1.0$, except that Hybrid A* has a objective function for the smoothing that also considers distance to obstacles. For the THA* algorithm, three cost maps were created by mapping the traversability map with three different $C_{\max}$ values, 2.0, 6.0 and 12.0.

In Fig. 3, we present the results from a single simulation, for which the quantitative data are listed in Table I. White represents areas of highest traversability, and a continuous scale shading towards black indicates lower traversability, with pure black representing obstacles. Several parameters are calculated for each of the paths. The accumulated traversability is calculated by (2), but using the normalized traversability map. We use the traversability map instead of the cost maps because the traversability map is the same for all the paths, whereas the cost maps are different for each of the $C_{\max}$ values and the accumulated costs are therefore not be directly comparable. The average traversability is the accumulated traversability divided by path length,. The average values over simulations in 10 different maps with a single trial in each are given in Table II, which also includes the standard deviations for the average traversability values. Table II shows that the regular Hybrid A* produces feasible paths that are shorter than the THA* paths; the THA* paths with $C_{\max} = 2.0$ are longer but with lower accumulated and average traversability, and the THA* paths with $C_{\max} = 6.0$ are even longer and with approximately the same accumulated traversability values, but with lower average traversability values. Increasing $C_{\max}$ to 12.0 actually increases both the path length and the accumulated traversability, and while the average traversability is lower than for $C_{\max} = 6.0$, the standard deviation is higher and the gain in average traversability is well within the standard deviation. This result illustrates how selecting cost value affects the path; by penalizing low traversable areas with higher cost value, the paths pass through more traversable areas as we see in Fig. 3 and the traversability values are improved. However, by increasing $C_{\max}$ to 12, the algorithm yields more unstable results with a negligible gain in average traversability.

The simulation results show that the THA* path planning method is capable of finding paths with improved traversability values compared to those found by the original Hybrid A* algorithm, in terms of both the average value over the length of the path and, in some (but not all) cases, the total accumulated traversability. This ability is somewhat affected by the selected cost scaling. When a higher cost is selected, the algorithm is more likely to avoid areas with low traversability as seen by the decrease in average traversability value.

In addition to evaluate the optimality of THA* with different values of $C_{\max}$, we conducted simulations to evaluate if the path-finding success rate of THA* is affected by the values of $C_{\max}$. We did this by increasing the amount of obstacles in the traversability maps generated by the Perlin noise until

Fig. 4. Olav, the autonomous UGV used in the experiment.

the algorithms started to fail to find collision free paths. Using 10 different traversability maps, the results did not show any differences in the path-finding success rate using Hybrid A* or THA* with $C_{\max}$ of 2.0, 6.0 or 12.0, they either all failed or all succeeded. This result indicates that the completeness of THA* is not affected by the value of $C_{\max}$.

IV. Real-Life Experiments

In this section, we present full-scale experiments conducted to verify and illustrate the proposed THA* path planner and compare the results of the proposed algorithm with those of the original Hybrid A* path planner.

A. Experimental Setup

The UGV used in the experiments, named Olav [30], is shown in Fig. 4. It is a 4WD Polaris Ranger 900XP modified for autonomous operation. The vehicle is Ackermann steered with a turning radius of approximately 4 m. When running in autonomous mode, we typically command a speed of 2.0 m/s and this speed was used in the experiments. A Velodyne HDL-32E LiDAR scanner and a Point Grey Grasshopper monochrome camera were mounted on the roof and used for perception. Two other cameras can be seen in the figure, but they were not used in these experiments. A high-accuracy inertial navigation system (INS) and global navigation satellite system (GNSS) module was installed, and a computer with an i7-7700K CPU and a GeForce GTX1080Ti GPU was used for perception processing and path planning.

For the experiments, we used a method for finding a traversability estimate based on the geometry of the terrain. First, a digital elevation model was created and maintained based on LiDAR measurements, using a method closely resembling the method described by [31]. The method gives both an elevation map and uncertainty estimates for the elevation values in the range of $[0.0, 1.0]$ for each cell where 1.0 means complete certainty of the height value. The map size was 80x80 m with a cell size of 0.25 m. Then, the elevation map was used to calculate a scalar inclination map based on the height difference between neighboring cells. The inclination values were then linearly mapped to a traversability map where 0° inclination was mapped to a traversability value of 0.0, and an inclination of 45° was mapped to a traversability value of 1.0. Everything above 45° inclination was considered as obstacles and thus set to a traversability value of 1.0 as well. 45° was chosen as a limit as this allows a maximum height difference between neighboring cells of 0.25 m given the cell size of 0.25 m, and we found that this was the limit for the size of the obstacles that our vehicle was able to pass due to its wheel size. Finally, the traversability values were weighted with the cell height uncertainty, so that uncertain cells were given lower traversability scores compared to cells with high certainty. These traversability maps were then used to calculate cost maps as described in Section II-C. For these experiments, the cost parameter $C_{\max}$ was set to 6.0, the same value as used for the green path shown in Fig. 3. In addition to the traversability map, the path planner uses the current pose estimate from the INS/GNSS system and a goal point provided by the operator.

The UGV can be operated using either the newly proposed THA* planner or the original Hybrid A* planner. With Hybrid A*, the traversability map is thresholded to create a binary obstacle/free-space map. Because the terrain is unstructured and difficult to classify correctly, incorrect classifications often occur. Therefore, a liberal threshold for what is considered free space is adopted to ensure that there will be few false positive obstacles that would bring the UGV to an unnecessary halt.

To account for the size of the vehicle, we performed a dilation of the traversability map before path planning, thus allowing the vehicle to be subsequently treated as a point mass during the planning process. The dilation size was set to 4 cells or 1.0 m, which is half the vehicle width (0.75 m) + 1 cell (0.25 m) as a safety margin. The kinematic center of the vehicle was selected as the reference point for path planning.

During autonomous driving, the path planner used a receding horizon strategy in which it planned a path as far towards the goal point as possible and then used this path for subsequent steering operations. The path planner was allotted 500 ms of processing time. After a fixed time period of 2 s, path planning was repeated based on the most recent map. When the goal point lay outside the current traversability map, a temporary goal point was used for path planning. This temporary goal point was located at the intersection of the edge of the map and a straight line between the vehicle and the goal point. If no path to this point was found within the allotted 500 ms computation time, the best path as evaluated by the heuristic was selected as the current path.

B. Experiments

The experiments were divided into two parts. In the first set of experiments, THA* was used by itself for autonomous navigation on rough terrain in various scenarios. In the second set of experiments, a single scenario was repeated several times using both Hybrid A* and THA*. For all experiments, the maps shown in Figs. 5 and 6 are the actual traversability maps generated during the live experiments with the same color scale as for the simulated case shown in Fig. 3.

1) Autonomous Driving Using Traversability Hybrid A:* In the first set of experiments, we used the THA* exclusively in long range scenarios. The purpose of these experiments was to evaluate if the THA* method is able to work in a real-life

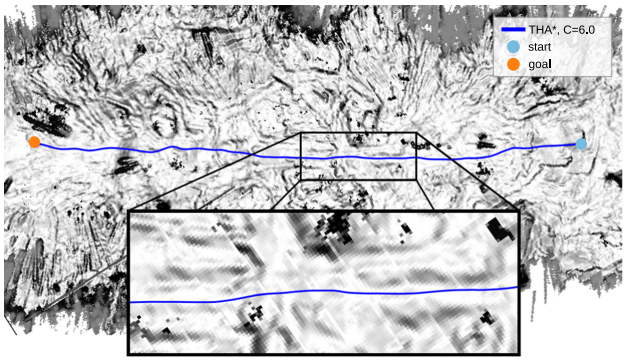

Fig. 5. Driven path on rough terrain using THA*. The driven distance was 153 m.

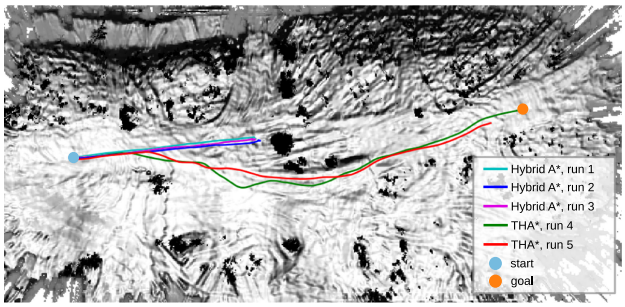

Fig. 6. Comparison of paths driven using the Hybrid A* and THA* algorithms. With Hybrid A*, the vehicle steered into a low-traversability area. With THA*, the vehicle found paths with higher traversability at the expense of a longer distance.

real-time scenario and generate paths that successfully takes the UGV to the desired goal. A total of eight routes were driven. The longest autonomously driven path was 278 m, and the average driven distance was 175 m. Fig. 5 shows the path driven in one of the eight cases. The path began as a relatively straight line from right to left, but small adjustments were made by the path planner to align it with the most traversable areas of the map. The UGV succeeded in navigating from the starting point to the goal point while simultaneously choosing a path with high traversability. In this case, no manual interventions were performed. This entire run can be seen in the accompanying video of this letter.

In Fig. 5 on the left side of the map, close to the end, the path moved through a gray area, i.e., an area with a low degree of traversability. Because there were no better alternatives close by, the path passed straight through. Overall, the map in this given case mostly has a high degree of traversability, and the resulting path between the starting and goal points is close to a straight line, which is as expected. Fig. 5 also shows a zoomed-in section of the path with a higher level of detail that shows how the path avoids gray areas insofar as possible.

Of the eight trial runs, five were completed without manual intervention. This is a strong result, considering that it was achieved over long distances on rough terrain about which the system had no prior knowledge. In two of the runs, the UGV became stuck in front of small plants that were classified as obstacles only at a close range without sufficient room for the UGV to avoid them. In these cases, a manual reset of the traversability map was performed. This reset cleared the map; then, the perception process was restarted with the construction of a new traversability map where the plants were correctly classified, and thus, the UGV resumed driving. In these cases, the path planning algorithm itself was not the problem; however, these examples show how sensitive the entire system is to traversability estimation errors. For the last run, the goal point was placed in the middle of a lightly vegetated area. In this case, the UGV was unable to find a path to the waypoint, despite several resets of the map.

The processing time required for the THA* algorithm also proved to be an issue. When sharp turns were planned, the path planning latency meant that the UGV continued to follow the old path for some time, moving away from the new path before it could react and steer onto the new path. This, in turn, led to the observation of oscillations in two trials. This issue resolved itself in these trials, as the UGV was able to detect large deviations between its current position and the path; then, the UGV intentionally stopped, which allowed the UGV and the path planner to synchronize, after which the UGV resumed by itself. Nevertheless, although these situations were handled without human intervention, this is still considered undesirable behavior.

2) Comparison of the Original Hybrid A and THA* Algorithms:* In the second set of experiments, THA* was compared with the original Hybrid A* algorithm by applying each of the two methods to the same driving case a total of five times. Fig. 6 illustrates the results of the five trials. The map is the traversability map produced by the perception system during the trial with the path shown in red. In all trials, the UGV started at the same position on the left and attempted to drive to the goal point shown at the end of the longest path.

The three shortest paths (blue, teal and magenta) were obtained using the original Hybrid A* method; these runs did not succeed in reaching the goal point. About halfway through these runs, the UGV entered an area with uneven surface and thus lower perceived traversability. The low traversability of the terrain manifested as rough motion of the vehicle in the pitch and roll directions. In the accompanying video, this is evident from the rougher motion of the camera compared to the smooth motion seen in the THA* run. In all the Hybrid A* trial runs, the UGV became physically stuck at the same spot where it encountered loose sand.

The two longer paths in Fig. 6 (red and green) were obtained from runs using THA*, in both of which the UGV succeeded in driving to the given waypoint. The red path is a few meters shorter than the green path; however, this difference arose only because the recording itself was stopped at this point. In the conducted experiment, the UGV actually reached the goal point just as in the other successful run. In the longest of the runs shown, the UGV drove 114 m. In both THA* cases, the UGV drove without any human intervention.

Comparing the two methods, it appears as THA* performed better than Hybrid A* as the UGV reached the goal with THA*. However, the UGV became stuck due to the loose sand it encountered and this is not modelled in the traversability calculation for THA* either. We could therefore not expect the THA* planner

to avoid this either if encountered. Thus, the fact that the UGV became stuck is not the main result of this experiment; rather, the rougher motion of the vehicle in the Hybrid A* runs is. Even if the UGV in the Hybrid A* runs had succeeded in traversing the rough area, the path through the more traversable area would have been preferred because it would lead to a smoother ride, resulting in higher quality of the payload sensor data and less wear and tear.

V. Conclusions and Future Work

In this letter, a new real-time path planning method for UGVs driving on terrain is presented. The method integrates traversability estimates directly into the planning process to optimize the planned path in terms of both traversability and distance. The proposed method was evaluated by means of both simulated terrain maps and real experiments with an autonomous UGV driving on unknown rough terrain. Eight live trials were conducted, and the results demonstrated that the new method was able to successfully steer the vehicle over distances of up to 270 m on rough terrain without manual intervention. In three of the trials, some manual intervention was required due to traversability estimation errors, but in two of those cases, the vehicle was able to recover and complete the mission. We also demonstrated that the new Traversability Hybrid A* algorithm was capable of finding paths through more traversable terrain than the Hybrid A* algorithm did in the same scenarios. The successful trials illustrate the usefulness of the proposed path planning method for autonomous driving on terrain. The implementation of the algorithm used in this demonstration revealed some issues related to the computation time that led to short planning horizons and, in rare cases, steering oscillations. In the future, this problem should be addressed. Work could also be done to incorporate more realistic traversability estimates based on the vehicle physics, which could further improve path planning in areas with low traversability.

References

[1] K. Iagnemma and S. Dubowski, *Mobile Roots in Rough Terrain - Estimation, Motion Planning, and Control With Application to Planetary Rovers, Ser.* Springer Tracts in Advanced Robotics (STAR). Berlin, Germany: Springer, 2004, vol. 12.

[2] G. Ishigami, K. Nagatani, and K. Yoshida, "Path planning for planetary exploration rovers and its evaluation base on wheel slip dynamics," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2007, pp. 2361–2366.

[3] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: A survey," *Eng. Appl. Artif. Intell.*, vol. 26, no. 4, pp. 1373–1385, 2013.

[4] C. Castejon, B. L. Boada, D. Blanco, and L. Moreno, "Traversable region modeling for outdoor navigation," *J. Intell. Robot. Syst.*, vol. 43, pp. 175–216, 2005.

[5] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Tran. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481-2495, Dec. 2017.

[6] F. Schilling, X. Chen, J. Folkesson, and P. Jensfelt, "Geometric and visual terrain classification for autonomous mobile navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2017, pp. 2678–2684.

[7] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1135–1145, Apr. 2016.

[8] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[9] D. Ferguson and A. Stentz, "The Field D* Algorithm for Improved Path Planning and Replanning in Uniform and Non-Uniform Cost Environments,"Carnegie Mellon University, Tech. Rep. CMU-TR-RI-05-19, 2005.

[10] J. Carsten, A. Rankin, D. Ferguson, and A. Stentz, "Global path planning on board the mars exploration rovers," in *Proc. IEEE Aerospace Conf.*, 2007, pp. 1–11.

[11] J. A. Sethian, "Theory, algorithms, and applications of level set methods for propagating interfaces," *Acta Numerica*, vol. 5, pp. 309–395, 1996.

[12] S. Garrido, L. E. Moreno, D. Blanco, and F. Martin, "FM2: A real-time fast marching sensor-based motion planner," in *Proc. IEEE/ASME Adv. Intell. Mech.*, 2007, pp. 1–6.

[13] S. Garrido, L. Moreno, F. Martin, and D. Alvarez, "Fast marching subjected to a vector field-path planning method for mars rovers," *Expert Syst. Appl.*, vol. 78, pp. 334–346, 2017.

[14] S. Garrido, D. Alvarez, F. Martí n, and L. Moreno, "An anisotropic fast marching method applied to path planning for mars rovers," *IEEE Aerosp. Electro. Syst. Mag.*, vol. 34, no. 7, pp. 6–17, Jul. 2019.

[15] C. Arismendi, D. álvarez, S. Garrido, and L. Moreno, "Nonholonomic motion planning using the fast marching square method," *Int. J. Adv. Robot. Syst.*, vol. 12, no. 5, 2015. [Online]. Available: https://journals.sagepub.com/doi/full/10.5772/60129

[16] V. González, C. A. Monje, L. Moreno, and C. Balaguer, "Fast marching square method for UAVs mission planning with consideration of dubins model constraints," in *Proc. 20th IFAC Symp. Automat. Control Aerosp.*, 2016, pp. 164–169.

[17] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–98, 1986.

[18] Q. Zhu, Y. Yan, and Z. Xing, "Robot path planning based on artificial potential field approach with simulated annealing," in *Proc. 6th Int. Conf. Intell. Syst. Desi. Appl.*, 2006, pp. 622–627.

[19] H.-T. Chiang, N. Malone, K. Lesser, M. Oishi, and L. Tapia, "Path-guided artificial potential fields with stochastic reachable sets for motion planning in highly dynamic environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2347–2354.

[20] S. M. LaValle, "Rapidly-exploring random trees: A. new tool for path planning," Computer Science Dept., Iowa State University, Tech. Rep. TR 98-11, 1998.

[21] O. Arslan, K. Berntorp, and P. Tsiotras, "Sampling-based algorithms for optimal motion planning using closed-loop prediction," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 4991–4996.

[22] R. Takemura and G. Ishigami, "Traversability-based RRT * for planetary rover path planning in rough terrain with LIDAR point cloud data," *J. Robot. Mechatron.*, vol. 29, no. 5, pp. 838–846, 2017.

[23] P. Wolf, T. Ropertz, M. Oswald, and K. Berns, "Local behavior-based navigation in rough off-road scenarios based on vehicle kinematics," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 719–724.

[24] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," in *Proc. 1st Int. Symp. Search Techn. Artif. Intell. Robot.*, 2008. [Online]. Available: https://ai.stanford.edu/~ddolgov/

[25] J. Petereit, T. Emter, C. W. Frey, T. Kopfstedt, and A. Beutel, "Application of hybrid A* to an autonomous mobile robot for path planning in unstructured outdoor environments," in *Proc. 7th German Conf. Robot.*, 2012, pp. 1–6.

[26] J. Jordan and A. Zell, "Real-time model based path planning for wheeled vehicles," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 5787–5792.

[27] B. Sebastian and P. Ben-Tzvi, "Physics based path planning for autonomous tracked vehicle in challenging terrain," *J. Intell. Robot. Syst.*, vol. 95, pp. 511–526, 2019.

[28] D. Belter, P. Labecki, and P. Skrzypczynski, "Adaptive motion planning for autonomous rough terrain traversal with a walking robot," *J. Field Robot.*, vol. 33, no. 3, pp. 337–370, 2015.

[29] K. Perlin, "An image synthesizer," in *SIGGRAPH '85 Proc. 12th Annual Conf. Comput. Graph. Interactive Techn.*, no. 3, 1985, pp. 287–296.

[30] K. Mathiassen, M. Baksaas, L. E. Olsen, M. Thoresen, and B. Tveit, "Development of an autonomous off-road vehicle for surveillance missions," in *Proc. IST-127/RSM-003 Specialists' Meeting Intell. Autonomy Robot.*, 2016, pp. 1–10.

[31] P. Fankhauser, M. Bloesch, C. Gehring, and M. Hutter, R. Siegwart, "Robot-centric elevation mapping with uncertainty estimates," in *Proc. Int. Conf. Climbing Walking Robot.*, Poznan, Poland, 2014.