Meta-Conv1D Energy-Aware Path Planner for Mobile Robots in Unstructured Terrains

Marco Visca 1, Roger Powell 1, Yang Gao 1, and Saber Fallah 1

1Affiliation not available

October 30, 2023

Abstract

Driving energy consumption plays a major role in the navigation of autonomous mobile robots in off-road scenarios. However, real-time constraints often limit the accuracy of the energy estimations, especially in scenarios where accurate wheel-terrain interactions are complex to model. This paper reports on first results of an adaptive deep meta-learning energy-aware path planner that can provide energy estimates of a mobile robot traversing complex uneven terrains with varying and unknown terrain properties. A novel feature of the method is the integration into the meta-learning framework of a 1D convolutional neural network to analyze the terrain sequentially, in the same temporal order as it would be experienced by the robot when moving, and efficiently adapt its energy estimates to the local terrain conditions based on a small number of local measurements. The performance of the method is assessed in a 3D-body dynamic simulator over several typologies of deformable terrains and unstructured geometries. We provide evidence of the benefit of the proposed approach to retain 83% r2 score of the original simulator at 0.55% of the computing time. Finally, we compare the method with alternative state-of-the-art deep learning solutions. In this way, we show indications of its improved robustness to provide more informed energy estimations and energy-efficient paths when navigating over challenging uneven terrains.
Meta-Conv1D Energy-Aware Path Planner for Mobile Robots in Unstructured Terrains

Marco Visca, Roger Powell, Yang Gao and Saber Fallah

Abstract—Driving energy consumption plays a major role in the navigation of autonomous mobile robots in off-road scenarios. However, real-time constraints often limit the accuracy of the energy estimations, especially in scenarios where accurate wheel-terrain interactions are complex to model. This paper reports on first results of an adaptive deep meta-learning energy-aware path planner that can provide energy estimates of a mobile robot traversing complex uneven terrains with varying and unknown terrain properties. A novel feature of the method is the integration into the meta-learning framework of a 1D convolutional neural network to analyze the terrain sequentially, in the same temporal order as it would be experienced by the robot when moving, and efficiently adapt its energy estimates to the local terrain conditions based on a small number of local measurements. The performance of the method is assessed in a 3D-body dynamic simulator over several typologies of deformable terrains and unstructured geometries. We provide evidence of the benefit of the proposed approach to retain 83 % r² score of the original simulator at 0.55 % of the computing time. Finally, we compare the method with alternative state-of-the-art deep learning solutions. In this way, we show indications of its improved robustness to provide more informed energy estimations and energy-efficient paths when navigating over challenging uneven terrains.

Index Terms—Autonomous mobile robots, deep learning, energy-aware path planning, machine learning, meta-learning

I. INTRODUCTION

Autonomous mobile robots (AMRs) can be of crucial importance to perform tasks in challenging off-road environments, such as search and rescue activities, nuclear plants’ inspections, agriculture, and planetary exploration. However, the driving energy consumption (i.e. the energy expenditure due to the traction of the wheels) often constrains the capability of a robot to operate autonomously [1]. Indeed, AMRs in off-road scenarios spend most of their time moving and have often limited on-board resources. AMRs that run frequently out of power must spend a longer time recharging their batteries and can travel shorter distances. Therefore, enabling AMRs with accurate driving energy prediction and planning can be beneficial to the success and efficiency of their operations.

In off-road applications, the main factors influencing the driving energy consumption are (1) the terramechanical properties and (2) the geometry of the terrain. The former denotes the mechanical characteristics of the terrain and its response to vehicular loading and shear stress [2]. Particularly, in natural environments, numerous types of terrains with different characteristics, ranging from sand, mud, and countless intraclasses, may be encountered. However, the main challenge with accurate driving energy predictions in those terrains is that the terramechanical properties are often a priori unknown, can constantly vary, and are cumbersome to estimate in real-time.

Concerning the geometry of the terrain, previous works have commonly approximated the terrain as a planar surface whose inclination is the only energy-relevant geometric factor [3]–[6]. However, off-road environments are often characterized by uneven surfaces such as scattered rocks, steps, bumps, and rough terrain. Traversing unstructured geometries can induce complex motion dynamics which pose a greater challenge to the locomotion and control system of the robot. For example, when stepping over rocks, external forces can act on the wheels and the wheels’ traction can sensibly change due to the variation of the wheel-ground contact area (with the extreme case being when the contact with one of the wheels is lost) [7]. Such complex interplay between varying terrain properties, unstructured geometries, and its effect on the driving energy consumption can be challenging to model and can demand an excessive computing workload for the implementation on board an AMR [8].

In this context, deep neural networks (DNNs) can be advantageous. DNNs have proved remarkable capabilities to efficiently extract features from high-dimensional data. Moreover, DNNs do not require explicit domain knowledge to be implemented into the prediction algorithm and can be efficiently integrated into parallel computing architectures [9]. Several studies have proposed the use of DNNs to improve the autonomy of robotic vehicles [10]–[12]. However, standard deep learning models commonly suffer from the need for a large labeled dataset and lack of generalization to unforeseen conditions. This makes the applications of DNNs non-trivial, in the context of wheel-terrain interaction for off-road robotics, due to the unknown and varying properties of the encountered terrains. Meta-learning offers promising solutions to address the aforementioned deep learning problems for its ability to enable rapid adaptation based on limited amounts of data [13].

In this paper,

1) We propose a deep-meta learning approach to learn and
adapt the driving energy prediction model of an AMR traversing terrains with (1) unknown and varying terramechanical properties and (2) complex unstructured geometries (Section III). The main novelty of our method is the development of a deep meta-learning architecture that exploits convolutional 1D (Conv1D) neural networks to analyze sequentially the full 3D geometry of the terrain. In this way, our algorithm better captures the complex wheel-terrain interaction over unstructured terrains from which the driving energy depends. Moreover, the meta-learning methodology alleviates the need of collecting in advance a large training dataset by allowing the energy estimator to efficiently adapt its predictions to new terrains based on a small number of local measurements.

2) We propose a method to integrate the adaptive energy estimator into a state-lattice A* path planner (Section III-D). The two approaches complement each other in that both energy estimation and path planning are performed directly over feasible trajectories, thus considering the actual robot mobility constraints.

3) We provide experimental results of the model performance in simulation over several typologies of natural terrains and unstructured geometries (Section IV) and we compare the proposed approach with alternative state-of-the-art deep learning solutions (Section V).

II. RELATED WORKS

Several works on robotic exploration of off-road scenarios have regarded the environment as made of a set of known terrain classes [10][11]. These methods commonly adopt terrain classifiers to determine the terrain class and LIDAR or stereo vision sensors to measure the terrain geometry. Hence, different energy models can be used, such as semi-empirical functions [3][6], look-up tables [4], or neural networks [5] to link the terrain geometry to the driving energy consumption for each terrain type. However, these methods do not account for terrains whose properties are unknown and they commonly consider the terrain inclination as the only energy-relevant geometric factor.

Existing wheel-terrain interaction simulators can already provide accurate estimates of AMRs’ contact forces, torques, drawbar pull, as well as driving energy consumption, in complex unstructured environments [14][15]. However, the computing workload of these simulators is often not suitable for use in a real-time optimization framework. Moreover, the simulators require the setting of terrain-dependent parameters that are often a priori unknown and difficult to estimate online [16][17].

Machine learning algorithms have been used to directly learn energy models from collected data. Martin et al. [18] proposed a gaussian process to build an energy map of a wheeled robot traversing an unknown terrain over various loops, while exploration was encouraged to converge to minimum energy tours of the environment. However, this method is impractical for robots that explore new terrains and do not perform repetitive loops. In [19], a deep learning algorithm is presented that exploits a 2D convolutional neural network to analyze RGB and depth images to determine the terrain driving energy consumption. However, the prediction of energy consumption from images assumes a strong correlation between the terrain visual appearance and its terramechanical properties. In many off-road scenarios, images can be misleading as terrains with similar visual characteristics may result, in contrast, in very different energy costs. Moreover, their method assumes a global energy model that requires prior training on a large and representative dataset while possible online adaptation to unforeseen conditions was not considered.

Meta-learning represents a promising approach to enable fast inference and adaptation based on limited amounts of data [13]. Other works have proposed the use of meta-learning for the online adaptation of robotic platforms [20][22]. In [23], an energy-aware deep meta-learning framework to predict, adapt, and plan over terrains with unknown and varying terramechanical properties was proposed. However, the prediction model considered the inclination of the terrain as the only important geometric factor for the estimation of the driving energy. Moreover, the training and test environments had even and uniform surfaces, while the effect of unstructured geometries on the driving energy consumption was not considered.

In [24], the use of Conv1D neural networks to model complex wheel-terrain interactions over unstructured geometries and their effect on the driving energy consumption was proposed. However, predictions were performed on a single terrain type with known terramechanical properties, while possible generalization and adaptation were not considered.

In this paper, we use similar implementations of [23] and [24] as main references for quantitative and qualitative comparisons. Specifically, we refer to them respectively as Meta-Plane and SingleTerrain-Conv1D (ST-Conv1D), while we refer to the method proposed in this paper as Meta-Conv1D.

III. METHODOLOGY

A. Geometric Analysis with Conv1D

We propose to analyze the terrain geometry end-to-end from point clouds via 1D convolutional neural networks. In this way, the sequential context of the wheel-terrain interaction can be captured by considering the actual traverse dynamics, thereby leading to better prediction accuracy when traversing highly uneven terrains. The point cloud is preprocessed according to the method described in [24]. A diagram of the procedure is illustrated in Fig. 1. In this approach, point clouds are downsampled into a 2D-voxel grid and rearranged in the same temporal order as they would be experienced by the AMR when traversing the terrain. Specifically, each row can be viewed as the terrain elevation under a section of the AMR along its width.
at a specific time, while each column illustrates the evolution of the terrain under a specific location of the AMR as it advances along the trajectory. For brevity of the paper, the details are referred to [24]. In our application, the voxel discretization is set to 6.25 cm × 6.25 cm, while each trajectory is 2.7 m long, resulting into a matrix $x$ of $110 \times 40$ elevation values. Then, a Conv1D neural network is used to analyze the terrain sequence. Fig. 2 illustrates the proposed architecture. It consists of 5 stacks of 1D convolutional layers with ReLU activation followed by 1D max-pooling and 1 stack of 1D convolutional layers with hyperbolic tangent activation followed by 1D average-pooling. All layers have kernel dimension $k = 3$ and zero padding and respectively 24, 32, 64, 64, 128, and 128 filters. Finally, the output of the network is flattened resulting in a 128-D vector $x_g$ that represents the latent geometric representation of the terrain.

B. Meta-Learning Preliminaries

Meta-learning is concerned with learning algorithms that can efficiently adapt to new tasks. Specifically, the goal of meta-learning is to exploit the similarity among the various past experience to learn a common prior for learning faster on new tasks. For this reason, meta-learning is also referred to as “learning to learn”. Formally, while in standard supervised learning the goal is to find the parameters $\theta$ of a function $f_\theta$ that minimizes a loss $\mathcal{L}$ over a single large dataset $D$ from a task, in meta-learning, the goal is to minimize the expected cost over a distribution $\rho(D)$ of several small datasets from different tasks. While different formulations of the meta-learning problem have been proposed [13], in this paper we opt for the meta-learning black-box approach that can be formally expressed as:

$$
\theta^* = \arg\min_\theta \mathbb{E}_{D=\{D^{tr}, D^{tr}\} \sim \rho(D)} \sum_{j=1}^{J} \left[ \mathcal{L}(f_\theta(x_j^{tr}, D^{tr}), y_j^{ts}) \right]
$$

(1)

Where $\rho(D)$ is a distribution over a collection of datasets $D$ from different tasks sharing some underlying similarity. Then, each dataset $D$ is divided into a meta-training dataset $D^{tr}$ and a meta-test dataset $D^{ts}$ both of them made of input-output pairs $(x, y)$ from the same task. Then, the meta-learning objective is to find the parameters $\theta$ of a function $f_\theta$ such that, for all the tasks drawn from the distribution $\rho$, given few (e.g. K) examples of a task $D^{tr} = \{(x, y)_i^{tr}\}_{i=1}^K$, we can successfully predict new pairs from the same task on the meta-test dataset $D^{ts} = \{(x, y)_i^{ts}\}$. In the black-box formulation, $f_\theta$ is a single neural network trained in an end-to-end fashion to predict $y^{ts}$ from $x^{ts}$ and $D^{tr}$. For more information on the meta-learning black-box formulation refer to [23].

C. Meta-Conv1D Energy Predictor

In this paper, we exploit the meta-learning framework to train an AMR to adapt its driving energy model to several terrains with unknown properties. Specifically, we assume that the geometric information $x$ (preprocessed with the method described in Section III-A) is the only available input feature $x^{ts}$ of the new terrain to perform the energy prediction $y^{ts}$. Conversely, no other property of the untraversed terrain is explicitly given to the prediction model, while it is treated as the unknown characteristic that varies across tasks. Hence, a few observations from the recent experience $D^{tr}$ are exploited to implicitly retrieve the unknown characteristics of the terrain and the information is leveraged to adapt the estimations.

1) Neural Network Architecture: Figure 3 shows our proposed neural network architecture. First, the 2D-voxel grid geometries of the meta-training dataset $x^{tr}_{i,1:K}$ (i.e. the previously traversed terrains) and of the meta-test dataset $x^{ts}_{i,1:K}$ (i.e. the new terrain to analyze) are fed to multiple instances of the Conv1D architecture described in Section III-A (having shared weights). The outputs are the latent meta-training and meta-test geometric representations $y^{tr}_{i,1:K}$ and $y^{ts}_{i,1:K}$. Then, $x^{tr}_{i,1:K}$ is concatenated to the respective driving energy consumptions $y^{tr}_{i,1:K}$ (measured by the AMR while traversing the terrains). Therefore, the meta-training dataset can be expressed as $D^{tr} = \{(x, y)_i^{tr}\}$. This is fed as input to a Long-Short Term Memory (LSTM) layer with 128 units. This choice is based on the reported abilities of neural networks with memory capacities of meta-learning [25]. In our application, each intermediate cell $i$ of the LSTM layer is presented with one example from $D^{ts}$. Hence, each LSTM cell implicitly captures the unknown characteristics $h_i$ of the terrain based on the $i$ examples provided up until that point and passes on its current estimate to the next cell. In meta-learning, the maximum number of meta-training examples $K$ is a heuristic application-dependent parameter. In our application, we experimentally set $K = 3$, as we observe minimal improvement in prediction error if more examples are provided. The top part of the neural network is a stack of two Fully Connected (FC) layers with relu and linear activations, and 64 and 1 units respectively. The FC stack (repeated $K$ times with shared weights) takes as input the LSTM hidden vector $h_i$ and the latent geometry of the meta-test terrain $x^{ts}_{i,1:K}$, and outputs the predicted energy $y^{ts}_{i,1:K}$ given the $i$ observations provided up to that point.

2) Training Procedure: Algorithm 1 illustrates the training procedure for the proposed neural network according to the black-box meta-learning framework. At each training step, for each element in a mini-batch, we randomly select a dataset $D_j$, from a collection of datasets each with different terramechanical properties. Then, $K$ meta-training samples $(x, y)^{tr}_i$ and 1 meta-
Algorithm 1 Meta-Learning Training Procedure

Input: Collection of datasets $\mathcal{D}$ from different terrain types
Input: Distribution $\rho$ over $\mathcal{D}$
Input: Mini-batch size $N$, number of examples $K$
Input: Neural Network $f_\theta$, learning rate $\alpha$

1: Randomly Initialize $\theta$
2: for $t=1,...$ do
3:    for $j=1,...,N$ do
4:        Sample $D_j \sim \rho(\mathcal{D})$
5:        Randomly sample $K$ $(x,y)^{tr} \sim D_j$
6:        Randomly sample $1$ $(x,y)^{ts} \sim D_j$
7:        $\hat{y}_{j,i}^{ts} = f_\theta([(x,y)^{tr}_i; x^{ts}])$
8:    end for
9:    $L = \frac{1}{NK} \sum_{j=1}^{N} \sum_{i=1}^{K} (y_j^{ts} - \hat{y}_{j,i}^{ts})^2$
10: $\theta \leftarrow \theta - \alpha \nabla_\theta L$
11: end for

Output: $\theta$

test sample $(x,y)^{ts}$ are randomly sampled from $D_j$. The energy predictions $\hat{y}_{j}^{ts}$ are obtained by feeding the samples to the network as described in Section III-C1. Hence, the loss for each batch is computed as in Line 9. Finally, the network weights are updated using stochastic gradient descent. In our experiments, the network is trained for 60 epochs, with mini-batch size $N = 32$, learning rate $\alpha = 10^{-4}$, and RMSprop optimizer [26].

3) Adaptation Procedure: After the training procedure, new observations from the local terrain can be exploited to adapt the model to the new terrain conditions. For example, if one observation is available, only the first LSTM cell must be filled in with $(x,y)^{ts}$. Hence, $h_1$ is used as input to the FC stack for the energy estimation of new terrain geometries $x^{ts}$.

D. Path Planning Integration

We perform path planning in a state-lattice space. State-lattice is a search graph where vertices representing kinematic states of the vehicle are connected by edges representing trajectories that satisfy its kinematic constraints [27]. Therefore, both cost estimation and planning are performed directly over feasible trajectories, considering the AMR mobility constraints. In this work, we define a set of 5 elementary trajectories 2.7 m long, and with curvature uniformly spaced in $[-0.13, 0.13]$ m$^{-1}$, based on the mobility capability of our robot (see Fig. 4a).

In our application, the cost of each edge in the state-lattice space must have a meaning of driving energy consumption. We propose a novel method to the aforementioned problem. A diagram of the procedure is described in Figure 4b. Any time a new planning procedure is started, recent observations from previously traversed trajectories are retrieved from memory and fed to the LSTM layer to recover the unknown properties $h$ of the terrain. Then, we assume the same properties $h$ for each upcoming edge in the graph. This can be considered as a reasonable assumption provided that replanning is performed often and that the terrain properties do not change with excessive frequency. Finally, the driving energy $y^{ts}$ of each edge $x^{ts}$ is estimated by: (1) retrieving and preprocessing the relative point cloud according to the method described in Section III-A, (2) concatenating the resulting latent geometry $x^{ts}_g$ to $h$, and (3) feeding the resulting vector to the FC stack of the model.

Finally, in line with the Meta-Plane and ST-Conv1D methods in [23] [24], we adopt the A* algorithm to optimize the path according to the driving energy consumption in the state-lattice space [28]. A*-like graph search methods have the advantage to guarantee bounded path optimality, and to reduce the computing cost of planning by making use of heuristics [29]. In this work, we adopt the same heuristic function $H$ as in [23] [24]. Specifically, we define $H(n,t) = \max\{0, (G\theta + \beta)d\}$, where $\theta$ and $d$ are respectively the relative inclination in degree and euclidean distance in meters between the current node $n$ and the target $t$, while $G$ and $\beta$ are heuristic parameters set to provide globally optimistic energy estimates for all available terrain types. In our application, $G$ and $\beta$ are set respectively to 1.41 kJ/m, and $-4$ kJ/m. In this way, the admissibility of the heuristic can be guaranteed for all terrains, thereby ensuring the convergence of the method to a predicted optimal solution.

IV. Simulator and Data Collection

The proposed method is tested using a dynamic simulator implemented in Python and based on the Chrono [30] implementation of the Soil Contact Model (SCM) [15]. SCM enables realistic modeling of arbitrary shaped wheel-terrain interactions in deformable terrains [31] [32] and retains a more efficient computing workload than alternative Finite Element or Discrete Element methods [8] [33]. Practically, terrains with different behaviors can be modeled in SCM by the setting of 6 terrain-dependent parameters: exponent of sinkage $n$, cohesive modulus $K_c$, frictional modulus $K_f$, cohesion limit $c$, angle of internal friction $\phi$, and Janosi-Hanamoto coefficient $J$ [2] [34]. In this work, 17 different terrain types are considered, whose SCM parameters are derived from [2] and [35]. The complete list with the corresponding SCM parameters is provided in Table I. In total, four macro-categories of terrain classes can be identified, each with considerably different terramechanical
TABLE I: Terrain Types and SCM Parameters.

<table>
<thead>
<tr>
<th>Category</th>
<th>Terrain Type</th>
<th>( n )</th>
<th>( K_c ) [kN m(^{-1})]</th>
<th>( K_\phi ) [kN m(^{-2})]</th>
<th>( c ) [kPa]</th>
<th>( \phi ) [deg]</th>
<th>( J ) [cm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. High</td>
<td>Clayey soil (Thailand, 38% wet)</td>
<td>0.5</td>
<td>13.19</td>
<td>692.15</td>
<td>4.14</td>
<td>13^o</td>
<td>2.0</td>
</tr>
<tr>
<td>Moisture</td>
<td>Clayey soil (Thailand, 55% wet)</td>
<td>0.7</td>
<td>16.03</td>
<td>1262.53</td>
<td>2.07</td>
<td>10^o</td>
<td>2.0</td>
</tr>
<tr>
<td>Content</td>
<td>Heavy clay (WES, 40% wet)</td>
<td>0.11</td>
<td>1.84</td>
<td>103.27</td>
<td>20.69</td>
<td>6.0^</td>
<td>3.0</td>
</tr>
<tr>
<td>Clay</td>
<td>Lean clay (WES, 32% wet)</td>
<td>0.15</td>
<td>1.52</td>
<td>119.61</td>
<td>13.79</td>
<td>11.0^</td>
<td>3.0</td>
</tr>
<tr>
<td>2. Loose</td>
<td>Dry sand (LLL)</td>
<td>1.1</td>
<td>0.99</td>
<td>1528.43</td>
<td>1.04</td>
<td>28.0^</td>
<td>2.0</td>
</tr>
<tr>
<td>Frictional</td>
<td>Sandy loam (Hanamoto, 26% wet)</td>
<td>0.3</td>
<td>2.79</td>
<td>141.11</td>
<td>13.79</td>
<td>22.0^</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Lunar Nominal Soil</td>
<td>1</td>
<td>1.40</td>
<td>820.00</td>
<td>0.17</td>
<td>35.0^</td>
<td>1.0</td>
</tr>
<tr>
<td>3. Compact</td>
<td>Sandy loam (LLL, 15% wet)</td>
<td>0.7</td>
<td>5.27</td>
<td>1515.04</td>
<td>1.72</td>
<td>29.0^</td>
<td>2.0</td>
</tr>
<tr>
<td>Frictional</td>
<td>Sandy loam (Michigan, 11% wet)</td>
<td>0.9</td>
<td>52.53</td>
<td>1127.97</td>
<td>4.83</td>
<td>20.0^</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Sandy loam (Michigan, 23% wet)</td>
<td>0.4</td>
<td>11.42</td>
<td>808.96</td>
<td>9.65</td>
<td>35.0^</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>LETE sand (Wong)</td>
<td>0.79</td>
<td>102.00</td>
<td>5301.00</td>
<td>1.30</td>
<td>31.1^</td>
<td>1.2</td>
</tr>
<tr>
<td></td>
<td>Upland sandy loam (Wong)</td>
<td>1.10</td>
<td>74.60</td>
<td>2080.00</td>
<td>3.30</td>
<td>33.0^</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Rubicon sandy loam (Wong)</td>
<td>0.66</td>
<td>6.90</td>
<td>752.00</td>
<td>3.70</td>
<td>29.8^</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>North Gower clayey loam (Wong)</td>
<td>0.73</td>
<td>41.60</td>
<td>2471.00</td>
<td>6.10</td>
<td>26.6^</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Grenville loam (Wong)</td>
<td>1.01</td>
<td>0.06</td>
<td>5880.00</td>
<td>3.10</td>
<td>29.8^</td>
<td>2.0</td>
</tr>
<tr>
<td>4. Dry</td>
<td>Heavy clay (WES, 25% wet)</td>
<td>0.13</td>
<td>12.70</td>
<td>1555.95</td>
<td>68.95</td>
<td>34.0^</td>
<td>0.6</td>
</tr>
<tr>
<td>Clay</td>
<td>Lean clay (WES, 22% wet)</td>
<td>0.2</td>
<td>16.43</td>
<td>1724.69</td>
<td>68.95</td>
<td>20.0^</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Steering: +0.07 Speed: +1.02 Steering: +0.06 Speed: +0.98 Steering: +0.02 Speed: +1.09 Steering: +0.01 Speed: +1.08
Braking: +0.00 Eng. Nm: +134.55 Braking: +0.00 Eng. Nm: +214.02 Braking: +0.00 Eng. Nm: +25.25 Braking: +32.96 Eng. Nm: +0.00

Fig. 5: Examples of generated maps with different terrain properties and unstructured obstacles in the Chrono simulator. In each example, from left to right, the terrain properties are set according to one of the terrain types from the macro-categories from 1 to 4 (described in Table I). In black, rigid unstructured obstacles are embedded in the maps. Colors on the wheel trace indicate the terrain sinkage depth after the AMR traversal (from dark blue being 0 cm to red being 0.15 cm and above). On the top left of each image, the onboard controller commands to the AMR at that snapshot time. On the top right, monitored metrics of interest at that snapshot time (i.e., the AMR longitudinal speed, the engine round per minute, and the engine torque).

properties. Moreover, different terrain types with more closely related properties can be found within each macro-category.

Figure 5 illustrates some examples of the generated environments in the Chrono simulator. The elevation maps are randomly generated with a Perline-noise algorithm, described in [36]. Then, the terramechanical properties of each map are set according to one of the 17 terrain types described in Table I. Finally, rigid, unstructured obstacles are embedded into the deformable terrains (in black). The robot model is the same as in [23]. The robot is subjected to a PID controller for the acceleration-brake pedals and a P controller for the wheel steering. Both controllers run at a frequency of 500 Hz.

The robot is driven over the generated environments at a constant speed of 1 m/s for a total of 1000 km (59 km for each of the 17 terrain types). This choice is motivated by the often low speed of robotic vehicles in natural environments [1] [18]. Then, the geometric features are collected with the method described in Section III-A. Meanwhile, the ground truth driving energy consumption is computed as:

\[
y = \sum_{t=t_0}^{t_f} \begin{cases} \tau_t \omega_t \Delta t & \tau_t \omega_t \geq 0 \\ 0 & \tau_t \omega_t < 0 \end{cases}
\]

(2)

Where \( \tau \) and \( \omega \) are respectively the engine torque and angular speed measured by simulated sensors on board the robot with a constant sampling rate \( \Delta t = 2 \text{ ms} \) over the time interval \([t_0, t_f] \) needed to traverse each segment. Note that Equation (2) is the mechanical formulation of driving energy consumption without considering loss components or energy recovery. Despite this being a simplified assumption, a detailed examination of specific engine characteristics is beyond the scope of this study. Moreover, as our method is entirely model-agnostic, this formulation can be easily adapted to any engine specifications. The final dataset is composed of 17 different datasets, each one from a different terrain type and made of 21 763 geometry-energy pairs.

V. RESULTS

A. Comparative Deep Learning Methods

This section provides a description of the alternative state-of-the-art deep learning approaches that are used to compare the prediction and planning performance of our method. All methods exploit the same training and validation datasets while they only differ in the network architecture and training procedure.

1) ST-Conv1D is a standard non-adaptive deep learning approach, proposed in [24]. The original method in [24] consisted of a Conv1D architecture trained on a single dataset from a known terrain type and without accounting for online adaptation.
In this paper, we devise a modification of [24] to allow its comparison with our meta-learning approach when they are tested on multiple unknown terrain types and provided with small numbers of previous observations. The first part of each ST model is composed of the same Conv1D architecture as Meta-Conv1D (as described in Section III-A). Then, the latent geometric representation $x_l$ is fed to 2 FC layers with relu activation and 16 and 8 units respectively, and 1 FC layer with linear activation and 1 unit. Therefore, upon training completion, each network can effectively model the geometry-energy relationship of its specific training terrain, but without any possibility to further adapt to new terrain types. For this reason, when testing on new terrain types, the previous observations are only used to select the most suited ST training model from the available collection. Specifically, the following approach is adopted: (1) K $(x_i, y_i)$ example pairs are observed from the terrain to test, (2) all the $x_i$ are fed to all the ST models, (3) the best model is chosen as the one having the most accurate prediction (in terms of mean squared error) compared with the true values $y_i$, and (4) the best ST model is used to predict the energy of new terrains.

2) Meta-Plane is the meta-learning approach described in [23]. The network exploits the same LSTM and FC layers of our approach as described in Section III-C. However, Meta-Plane differs from Meta-Conv1D in that the pitch and roll inclinations of the terrain (estimated with standard geometric features extraction techniques) are the only geometric features considered for the energy estimation.

All methods are tested on an Intel Core i9-9940X CPU and a GeForce RTX 2080 Ti GPU. Evaluation metrics for the comparisons are (1), for the prediction performance, the root mean squared error (RMSE) and the $r^2$ score (R2) between the predicted and true energy data and (2), for the planning performance, the average node time (i.e., the average time needed to expand a node in the state-lattice graph), the total planning time, and the real driving energy consumption.

### B. Generalization Performance

In this section, we compare the generalization performance of the three deep learning methods, prior to their integration into the path planner. Specifically, we perform 5 test trials with different training-validation splits. For each trial, the training dataset is constituted by 4 randomly selected terrain types, and the corresponding datasets, each one of them coming from one of the 4 macro-categories described in Section IV. Hence, the remaining 13 datasets are used for validation to assess the adaptation performance to new terrain types.

Table II summarises the adaptation performance of the three methods, averaged over the 5 test trials, and for different numbers of meta-training observations (columns from 1 to 3). Particularly, the methods are assessed over three different subsets of the validation datasets: (1) Mixed is the full set of validation datasets, (2) Unstructured is the subset that contains unstructured geometries, and (3) Simple is the subset that does not contain unstructured geometries. We observe that all the methods demonstrate the capability to improve their predictions as more meta-training observations are given (RMSE decreases and R2 increases). However, the two meta-learning approaches (i.e., Meta-Plane and Meta-Conv1D) retain consistently better performance. This can be explained by that ST-Conv1D relies entirely on the similarity between the training and validation datasets, without accounting for adaptation. Therefore, while ST-Conv1D can be suitable for environments with known terrain classes, it fails to generalize to unforeseen terrain types. Conversely, the meta-learning methodology enables an efficient adaptation based on the previous observations, thereby leading to better generalization to previously unknown scenarios. Furthermore, the prediction performance of all the methods degrades when analyzing terrains with complex unstructured geometries. Nevertheless, Meta-Conv1D demonstrates to retain better performance in all scenarios. Particularly, the advantages of Meta-Conv1D are more evident in the Unstructured subset. For example, the R2 with 1, 2, and 3 meta-training observations of Meta-Plane and Meta-Conv1D are of respectively [62.40%, 68.33%, 70.55%] and [68.27%, 73.48%, 75.10%]. Therefore, the prediction improvements of Meta-Conv1D over Meta-Plane are of respectively 5.87%, 5.15%, 4.55% in the Unstructured subset. Meanwhile, the improvements in the Simple subset are more modest, being of respectively 1.55%, 0.66%, 0.30%. This provides evidence of the benefit of the proposed method to more accurately capture the wheel-terrain interaction by using Conv1D neural networks, thereby resulting in more accurate driving energy estimates in complex unstructured scenarios.

### C. Path Planning Integration

In this section, we compare the performance of the three energy predictors integrated into the A* state-lattice energy-aware path planner. Specifically, the three methods are tested on randomly generated new scenarios with unstructured geometries and unforeseen terrain properties (selected from the 13 validation terrain types). Moreover, to further increase the uncertainty of the terrain properties encountered during testing, random noise is added to each one of the 6 terramechanical parameters (with noise bounded to ±5% of the value of each parameter).

The performance of the three methods, integrated into the A* path planner, are compared by conducting statistical analysis over 1080 random-start goal positions within different test environments. For each start-goal position: (1) the vehicle navigates for 8.1 m and collects the three most recent geometry-energy pairs, and (2) the three example pairs are used by the alternative
methods to plan a path to the next target. In this way, the performance of the three algorithms can be tested under identical initial conditions (i.e. both start-goal, and previous example pairs). Moreover, the number of unstructured geometries in the test environments is progressively increased after each 120 executions (from 0 to 0.8 \( n_{ug}/m^2 \), where \( n_{ug} \) is the number of unstructured geometries). In this way, the impact of the unstructured geometries on the prediction performance of the three methods can be analyzed.

Figure 6 summarises our findings. As expected, Meta-Plane and Meta-Conv1D have similar prediction performance over planar terrains (i.e., 0 \( n_{ug}/m^2 \)), with R2 of respectively 92.07 % and 91.07 %. Therefore, both methods can accurately adapt their energy prediction models to unknown terrain types if no rigid unstructured geometry is present. However, the prediction performance of Meta-Plane more rapidly degrades as the roughness of the environments increases, while Meta-Conv1D retains consistently better performance. This provides evidence of the increased robustness of Meta-Conv1D to better adapt to highly unstructured scenarios. Meanwhile, ST-Conv1D has consistently lower performance than Meta-Conv1D in all scenarios and lower performance than Meta-Plane for \( n_{ug}/m^2 < 0.6 \). This can be explained by the inability of ST-Conv1D to adapt its predictions to unforeseen terrain properties. However, ST-Conv1D has better prediction performance than Meta-Plane for \( n_{ug}/m^2 > 0.6 \). This can be interpreted as that, in particularly rough terrains, the benefit of analyzing the terrain geometry based on Conv1D neural networks, even without adaptation, can be greater than the benefit provided by the meta-learning adaptation without the Conv1D analysis.

In Table III, the overall prediction and planning performance over the 1080 traverses is illustrated. We observe that Meta-Plane has the longest average node expansion time and total planning time. This can be explained by that, in our implementation, the pitch and roll geometric feature extraction procedure takes marginally longer than the point cloud preprocessing into a 2D voxel-grid (described in Section III-A). Conversely, ST-Conv1D has the shortest node expansion time and total planning time. This can be explained by that, upon the choice of the best ST model from the ST-Conv1D collection (done only once at the beginning of each planning procedure), this approach involves the feed-forward computation of a much simpler deep learning model. Finally, Meta-Conv1D has intermediate time performance. Nevertheless, the node time of all the methods is considerably lower than the computing time of the SCM 3D-body dynamic simulator (see Section IV). Specifically, the expansion of one node using the SCM simulator is measured to take, in our application, around 9.8 s. Therefore, the computing time of the proposed Meta-Conv1D approach is about 0.55 % of the SCM dynamic simulator. Moreover, Meta-Conv1D provides the solutions that result in the lowest real driving energy consumption and with considerably higher prediction performance.

Fig. 7 illustrates a qualitative example of the planned paths using the three deep learning methods. On the bottom right, the recent experience (i.e. the terrain point clouds and the corresponding energy consumptions) from the three previous trajectories is shown. Hence, the past observations are differently preprocessed and analyzed by the three deep learning methods to plan an energy-aware path to the next target. As with previous results, ST-Conv1D has the shortest planning time but also provides the most energy-demanding solution. Meanwhile, Meta-Plane has the longest planning time and provides the least accurate prediction. Finally, Meta-Conv1D provides the solution with the lowest driving energy consumption and the most accurate prediction accuracy. This confirms the higher robustness of our method that, by both (1) adapting the prediction model based on previous observations and (2) analyzing the full 3D geometry of the terrain using Conv1D neural networks, can provide more informed energy estimations, thereby resulting in more energy-efficient paths in highly unstructured environments.

VI. CONCLUSION

In this study, we presented an adaptive deep meta-learning energy-aware path planner for AMRs navigating in challeng-
ing off-road terrains. We remark the benefit of the proposed method to (1) efficiently adapt its energy estimates to unknown scenarios based on small numbers of recent observations, (2) to retain similar performance to a realistic 3D-body dynamic simulator while sensibly reducing the required computing time, and (3) to provide more accurate driving energy estimations and energy-efficient paths than alternative state-of-the-art deep learning methods. Therefore, our approach can be of interest to several applications in the context of autonomous exploration of unknown challenging environments, for which an incorrect estimation of the driving energy consumption can compromise the success and efficiency of the mission.

We are extending this work in several directions. While in the current implementation, single-point energy estimates were performed, the data collection process may be affected by great uncertainty while traversing highly unstructured terrains (due to the increased localization and control challenges). To better address the uncertainty in the data, future works may consider a probabilistic extension of the meta-learning energy prediction and planning framework. Finally, while in this paper adaptation to new terrains was tested in simulation, exploiting the SCM 3D-body dynamic simulator, real-world tests will be required to further validate the results obtained during simulation.

REFERENCES

[13] H. Peng, “A comprehensive overview and survey of recent advances in 3D-body dynamic simulator, real-world tests will be required and planning framework. Finally, while in this paper adaptation to new terrains was tested in simulation, exploiting the SCM 3D-body dynamic simulator, real-world tests will be required to further validate the results obtained during simulation.

REFERENCES