IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Deep Reinforcement Learning for Safe Local Planning of a Ground Vehicle in Unknown Rough Terrain

Shirel Josef and Amir Degani, *Member, IEEE*

***Abstract*— Safe unmanned ground vehicle navigation in unknown rough terrain is crucial for various tasks such as exploration, search and rescue and agriculture. Offline global planning is often not possible when operating in harsh, unknown environments, and therefore, online local planning must be used. Most online rough terrain local planners require heavy computational resources, used for optimal trajectory searching and estimating vehicle orientation in positions within the range of the sensors. In this work, we present a deep reinforcement learning approach for local planning in unknown rough terrain with zero-range to local-range sensing, achieving superior results compared to potential fields or local motion planning search spaces methods. Our approach includes reward shaping which provides a dense reward signal. We incorporate self-attention modules into our deep reinforcement learning architecture in order to increase the explainability of the learnt policy. The attention modules provide insight regarding the relative importance of sensed inputs during training and planning. We extend and validate our approach in a dynamic simulation, demonstrating successful safe local planning in environments with a continuous terrain and a variety of discrete obstacles. By adding the geometric transformation between two successive timesteps and the corresponding action as inputs, our architecture is able to navigate on surfaces with different levels of friction.**

I. Introduction

Navigation in unknown rough terrain is necessary in many applications, from planetary exploration and mapping to search and rescue missions. In these applications, the high level of uncertainty can produce unexpected situations that can pose danger to the human operator. Therefore, unmanned navigation can contribute to completing these missions without threatening human lives.

Rough terrain can introduce a wide variety of obstacles that require different maneuvers as illustrated in Fig. 1. Hence, a crucial aspect of unmanned navigation is navigating the Unmanned Ground Vehicle (UGV) from a start position to a goal position while ensuring the safety of the UGV and its contents.

Extensive work has been done in the field of motion planning; for example, grid search motion planners and sampling-based motion planners such as rapidly exploring trees [1]. Other notable methods for motion planning include the dynamic window approach [2], potential functions [3], the vector field histogram approach [4], and bearing-only navigation [5]. However, unstructured outdoor environments pose a challenge to classic motion planners, as the UGV must operate in a dynamic environment while avoiding a vast range of obstacle types.

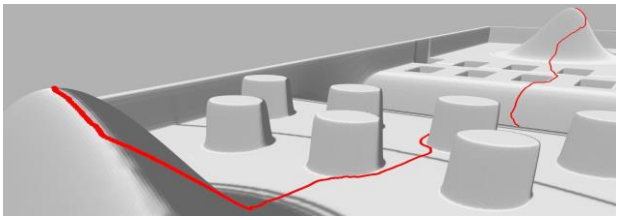


Figure 1. Safe path planning in a complex environment generated by our deep reinforcement learning approach. The simulated environment incorporates various obstacles that can be encountered in rough terrain such as: steep hills, ditches, unclimbable boulders and pits.

Initial work addressed rough terrain navigation by binary classification of the terrain as obstacles or free space, and applying an extension of the tangent bug algorithm [6]. Combination of binary motion planners and traversability classification was also proposed in [7] by combining RRT and gradually increasing the threshold of obstacle classification. While the previous methods might be too conservative, classifying a certain obstacle as non-traversable can lead to a less optimal plan or even failure to find one. Due to rough terrain's nature, such an obstacle might be traversable from a certain direction or at a specific velocity [8]. Therefore, in rough terrain navigation it is not clear how to binarize the environment, and a continuous approach that weighs the cost of traversing the terrain should be considered.

State lattice was among several works that introduced the concept of continuous obstacles and assigning cost to trajectories, and is described in [9]. A state lattice, which represents the UGV's motion, is searched using the A* algorithm with respect to the cost map. While a grid search over a cost map provides completeness and optimality depending on the search space level of discretization, it does require prior knowledge of the terrain and its.

In an unknown rough terrain scenario, global path planning might not be possible due to GPS deprivation, limited sensing, and a partially known or dynamic environment. Planning a global path might require multiple corrections and recomputations in the presence of newly acquired data; therefore, local planning is preferred.

Local navigation in unknown rough terrain is composed of two parts. The first part is planning a possible path toward the target position based on the UGV's kinematic constraints and surface geometry. The second part is tracking the selected path while considering the UGV's interaction with the terrain [10].

Traditional approaches to local navigation in rough terrain

S. Josef is with the Technion Autonomous Systems Program, Technion – Israel Institute of Technology, Haifa 3200003 ISRAEL (Email: shirel@campus.technion.ac.il).

A. Degani is with the Faculty of Civil and Environmental Engineering and with the Technion Autonomous Systems Program, Technion – Israel Institute of Technology, Haifa 3200003 ISRAEL (Email: adegani@technion.ac.il).

include selecting a path that minimizes the risk and effort of traversing the terrain from a set of candidate paths [8].

Various adaptations of the mentioned methods were introduced to address rough terrain motion planning. The potential field methods were extended to local sensing navigation on uneven terrain with binary obstacles in [11], continuous obstacles in [12], and near-optimal navigation by incorporating a high level global planner [13]. Combining the dynamic window approach [2] with potential functions [3] lead to a convergent dynamic window approach to obstacle avoidance for flat terrain [14] and rough terrain [15]. The vector field histogram method was extended to a traversability field histogram to determine the obstacle density within the range of the sensors [16].

In unknown rough terrain navigation, it is crucial to react quickly to sensor-gathered information. However, the computational complexity of existing motion planning methods increases with the complexity of the navigation problem. This issue is more acute in rough terrain, as we wish to analyze the traversability in a continuous manner. As Lacaze et al. predicted, current motion planners learn from experience to improve their ability to find correct paths [8]. Motion planning networks have the ability to learn, in a supervised approach [17] and reinforcement approach [18], to map state to action in order to navigate from a start state to an end state in a binary workspace configuration. Also, motion planning networks require a constant inference time and have a low memory footprint. Zhang et al. used Deep Reinforcement Learning (DRL) for unknown rough terrain navigation [19]. The work did not consider limiting the robot's orientation to ensure safety and exhibited a binary obstacle motion planning, e.g., circumnavigating hills that pose a problem when a goal is positioned on a hill.

One concern with autonomous vehicles is the human trust issue as it is sometimes difficult to explain how the autonomous system arrived at a specific decision. One way of building this trust is comprehensive sensing where it is clear that the UGV is aware of its surrounding and focuses on relevant information. The attention mechanism combined with reinforcement learning as showed in [20] allows to add interpretability to the decision process by visualizing where the agent is focusing on. However, their approach relies on using frames and recurrent cells' hidden states as input to the attention module and is therefore restricted to recurrent models. Manchin et al. addressed this issue by using a self-supervised approach where the input to the attention is only frames [21]. UGV's navigation requires several sensors and is not limited to spatial-visual sensors, therefore, a more general attention approach should be taken.

In this paper, we present a DRL approach to generating local paths toward goal positions in unknown rough terrain environments while maintaining the UGV's safety. We propose and examine four input types depending on the UGV's available sensors. After the suitable input is chosen, it is fed into a Rainbow based architecture [22]. In order to evaluate our approach, we use two baselines. Baseline-1 is inspired by potential functions and bearing only navigation, and baseline-2 is based on ego-graphs and local motion planning search spaces [23]. Our approach is less computationally demanding during the online path finding stage, while providing superior results compared to local planning baselines. We validate our approach in a dynamic simulation and show that it successfully manages to capture the vehicle's dynamics and interaction with the terrain while traversing continuous terrain and avoiding a variety of discrete obstacles.

The paper is organized as follows: In Sec. II, we provide a brief background on reinforcement learning. Next, in Sec. III, we describe in detail our deep reinforcement learning solution. In Sec. IV, we present kinematic simulation results compared to previous baseline motion planners. In Sec. V, we extend and validate our approach in a dynamic simulation. Section VI concludes the paper.

II. BACKGROUND

Reinforcement Learning (RL) is concerned with how to map situations to actions by interacting with the environment and maximizing the cumulative reward received from the environment [24]. One key feature of RL is considering the problem of a goal-directed agent interacting with an uncertain environment, which is crucial for navigation in unknown rough terrain.

A. Markov Decision Process

Interaction between an agent and an environment is formalized as a Markov Decision Process (MDP). At each time-step ($t = 0,1,2,3 \ldots$) the agent receives an observation of the environment's state, S_t, and acts accordingly by choosing an action, A_t. In the next time-step, the agent receives a scalar reward signal, R_t, from the environment and a new observation, S_{t+1}, which is determined by the environment dynamics, $p(s_{t+1}|s_t, a_t)$. The discounted return $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$, is the discounted sum of future rewards. The discount rate, $0 \leq \gamma \leq 1$, determines how myopic the agent is. Given the environment's state, the agent selects an action according to probabilities mapped by a policy, $\pi(a|s)$. The agent aims to learn such a policy, π, which maximizes the expected discounted return. Estimating the expected return starting from state s, taking action a, and following policy π is defined as $q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$ and called the action-value function. The agent explores the environment to approximate the optimal action-value function, denoted as q^* and defined as $q^*(s,a) = \max_\pi q_\pi(s,a)$. If the state and action space are small, the action-value function can be represented by a lookup table or a linear or nonlinear parametrized function.

B. Deep Q-Network

In real-world problems such as rough terrain navigation, the state space's large size makes it infeasible to store and learn the q-value estimations for each state-action pair independently. To address this issue, Deep Q-Network (DQN) used deep convolutional neural networks to approximate the optimal action-value function, $q^*(s,a)$, and successfully learned to map high-dimensional data (pixel frames of Atari games) to discrete actions [25]. Furthermore, the problem of instability or divergence in RL, when used with a nonlinear function approximator such as a neural network, is addressed in DQN.

C. Rainbow

Recent developments in RL can be integrated into DQN to improve action-value function estimation, increase sample efficiency, deal with sparse rewards, and perform more

robust exploration. The integrated agent, called Rainbow, outperformed DQN and achieved state-of-the-art performance [22]. Our DRL approach will use the above-mentioned architecture.

D. Attention and Self-Attention

The attention mechanism allows to model the dependencies between two sources by computing a compatibility score. As described in [26], an attention function maps a query and a set of key-value pairs to an output corresponding to the relative importance of each value to the query. Given a key $x_i \in \boldsymbol{x} = [x_1, x_2, \dots, x_n]$ and a query q a compatibility function $g(x_i, q)$ computes the compatibility score. Each value is assigned a weight computed by the compatibility function of the query and the complementary key. The softmax function is applied to the scores vector $[g(x_i, q)]_{i=1}^{n}$ to produce normalized weights. Each weight indicates the importance of the value to the query q. One of the most common attention function is the concat attention [27], defined as $g(x_i, q) = w^T \tanh\big(W(x_i; q)\big)$. In self-attention, which is a special case of attention, the query and the key-value pair both come from the same source.

III. Deep Reinforcement Learning for Safe Local Planning in Unknown Rough Terrain

To safely navigate in unknown rough terrain, we present several end-to-end DRL architectures with respect to the UGV's available local-range sensing.

A. Problem Formulation

The UGV has a three-dimensional position (x, y, z) and a three-dimensional orientation $(yaw, pitch, roll)$. The value of z is determined by the UGV's (x, y) position on the terrain map, and the $(pitch, roll)$ values are determined by (x, y, yaw). Therefore, (x, y, yaw) can be used to describe the UGV's configuration in the environment. We will refer to (x, y, yaw) configuration as a position. Our work considers a simple kinematic model which consists of three actions - forward, right, and left - and assumes that changes in position are small enough compared to changes in the terrain. A path is composed of a start position, q_{start}, a goal position, q_{goal}, and a set of positions, Q_{path}, generated by the constrained motion imposed by the kinematic model. If for all $q \in \{q_{start}, Q_{path}, q_{goal}\}$ it holds that the UGV's $roll < 60°$ and $pitch < 60°$, then the path is considered safe. Our goal is to construct a safe path from q_{start} to q_{goal}. The roll and pitch thresholds are based on an approximation of the UGV's geometric center of gravity and constraints of the vehicle. It is possible to choose any other thresholds depending on the UGV's properties and desired safety margins.

Dealing with local planning in unknown rough terrain, the agent has no access to a global map. Additionally, regarding the goal position, the only available knowledge is the angle to the goal relative to the UGV's heading.

B. Terrain Generation

In rough terrain navigation, it is important to minimize roll and pitch to avoid flips (pitch direction) and rollovers (roll direction). Therefore, to reach the peak of a steep hill without reaching high roll or high pitch, it should be climbed in a spiral fashion, which is controlled by the UGV's yaw. To simulate these situations, we combined position-randomized gaussians, such that their overlap created a unique mixture of hills and valleys, and formed a continuous terrain. Although the generated terrain is continuous, our dynamic validation shows that our method can also be successfully generalized to environments with continuous terrain and different types of discrete binary obstacles. The elevation map is 200x200 m^2, with cell size of 0.025x0.025 m^2.

C. Zero-Range Sensing Input

Limited sensing can arise from operating in hazardous environments or low production cost requirements. We define zero-range sensing as only having interoceptive sensing capability, such as Inertial Measurement Unit (IMU), and lacking any exteroceptive sensors such as laser scanners or depth cameras. Assuming zero range sensing, the UGV is only equipped with an IMU and can also sense the angle to q_{goal}. We concatenate roll, pitch, and relative angle to q_{goal} values into an input vector, as shown in Fig. 2a.

D. Immediate-Range Sensing Input

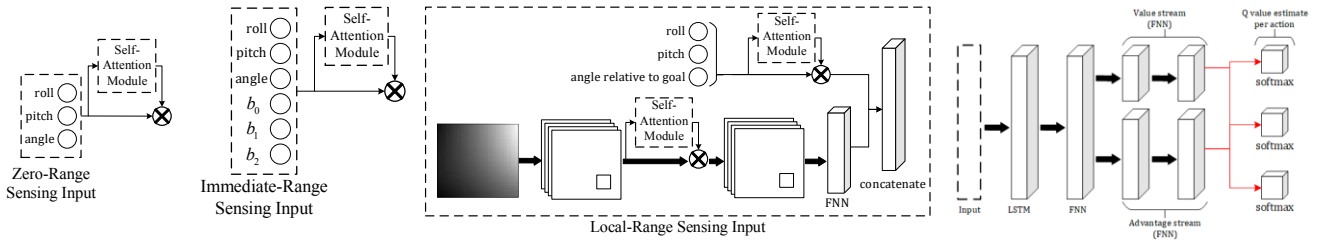
We assume a case in which the sensors can only estimate a one-step look-ahead of a position that can be reached by executing a single action. According to the terrain estimation at that position, we produce a binary value, b_i, such that $\{b_i | b_i \in \{0,1\}, i \in \{0,1, \dots, |Actions| - 1\}\}$. For each action i, b_i indicates whether or not that action results in a traversable position, receiving a value of 1 or 0 respectively. A binary signal was used, as it may be easier and faster to learn, similar to [28]. In future work, expert knowledge can be incorporated by extending the binary signals to continuous signals, describing levels of risk. The binary inputs are then concatenated with the UGV's current roll, pitch, and relative angle to q_{goal}, into an input vector, as shown in Fig. 2b.

E. Local-Range Sensing Input

When a broader range sensing capability is available, a local elevation map along with the interoceptive sensing, can be fused as the input. The 3.2x3.2 m^2 elevation map is fed into convolutional layers, followed by fully connected layers. The output of this branch is concatenated with the UGV's current roll, pitch, and relative angle to q_{goal}. The concatenation serves as an input to the network, as shown in Fig. 2c. A local elevation map can be centered around the UGV or cover the area that lies in front of the UGV. These two configurations are used in our experiments.

F. Self-Attention Modules

The importance of an input x_i with respect to all inputs $\boldsymbol{x}$ is achieved by taking $\boldsymbol{x}$ as the query and x_i as the key. Following the concat attention function $g(x_i, q) = w^T \tanh\big(W(x_i; q)\big)$, we argue that $W(\boldsymbol{x})$ yields the same importance as $W'(x_i; \boldsymbol{x})$ as no new information is introduced in the later. For non-spatial input, i.e., roll, pitch, and relative angle to q_{goal}, we feed the input to a fully connected layer followed by $\tanh$ activation. The result is inserted to a fully connected layer followed by sigmoid activation. The weights produced and the input are combined through element-wise multiplication. A similar process is performed to the spatial input, i.e., local height map, with the addition of applying a



(a) (b) (c) (d)

Figure 2. (a-c) Examined input types for DRL method. (d) Rainbow network architecture.

1x1 convolution to the input and reshaping the softmax output for element-wise multiplication. The self-attention modules are presented in Fig. 3.

G. Network Architecture

Our network architecture and training are based on a Rainbow agent, depicted in Fig. 2d. After the suitable input is chosen, it is fed into a Long Short-Term Memory (LSTM) layer. In local rough terrain navigation, our observation is only a partial description of the system's state. For example, given roll, pitch, and relative angle to q_{goal}, it is not possible to infer whether the vehicle is ascending a hill or making its way out of a valley. LSTM allows estimating the underlying system's state, as was presented in [29]. The LSTM's output is inserted to a fully-connected layer, and its output is split into an advantage stream and a value stream, as in Rainbow's dueling networks [30]. The output of the network is a distribution of the q-value estimate for each state-action pair, which can be used to calculate the mean q-value estimate, as explained in [31]. The original Rainbow agent, described in [22], replaced all fully connected layers with Noisy Networks. However, in our experiments, Noisy Network did not yield significant improvement compared to ε-greedy, as was also shown in [32]. Therefore, Noisy Networks were omitted from our architecture.

H. Reward Function

One common issue with reward functions for goal-directed agents is exploiting the reward in a positive reward loop. For example, the agent presented in [33] learned to drive in circles because no punishment was given for driving away from the goal. We address this issue by constructing a reward function inspired by the structure of a round shooting target. The goal position, q_{goal}, is encircled by n evenly spaced concentric rings. Every time the agent enters an inner ring, it receives a positive reward, R_{ring}. However, if the agent exits the ring, the reward, R_{ring}, is deducted. Rings found in proximity to the target indicate a higher reward to be received or deducted, and the amount of reward is controlled by the parameter, α. Upon reaching an ε-environment around q_{goal}, X_ε, the agent receives a positive reward signal, R_{goal}. Behaviors resulting in high roll or high pitch can be discouraged by presenting a negative reward to the agent. While this may be true, due to empirically sufficient results, if the UGV exceeds roll or pitch threshold values, the episode terminates and the reward given to the agent is zero.

The reward, $\mathrm{R_t}$, is given at each time-step after executing an action, and formalized as follows:

$$\mathrm{R_t} = \begin{cases} \alpha \cdot R_{ring}, & \text{enters a ring} \\ -\alpha \cdot R_{ring}, & \text{exists a ring} \\ R_{goal}, & \text{enters } \mathrm{X_\varepsilon} \\ 0, & \text{elsewhere} \end{cases} . \quad (1)$$

I. Model Training and Evaluation

Four models were trained and evaluated: DRL zero-range sensing, DRL immediate-range sensing, DRL local-range sensing with a 3 m box centered around the UGV, and DRL local-range sensing with a 3 m box in front of the UGV. Our models were trained in an episodic setting. For each episode, we randomly selected feasible start and goal positions $\{q_{start}, q_{goal}\}$ within our 200x200 $\mathrm{m^2}$ generated terrain. The agent interacts with the environment using our kinematic model according to ε-greedy exploration. When either the goal is reached or roll/pitch values exceed unsafe configuration values, the episode terminates. Each episode was limited to 1,500 time-steps and a maximum number of 100,000 episodes was allowed.

The models were evaluated every 10,000 episodes. The best parameters were chosen based on 100 randomly selected start and goal evaluation positions. The evaluation process is adopted to avoid over-fitting to the start and goal test positions. We used a 3.40GHz × 8, Intel Core i7 processor, with 32 GB RAM for model training and evaluation.

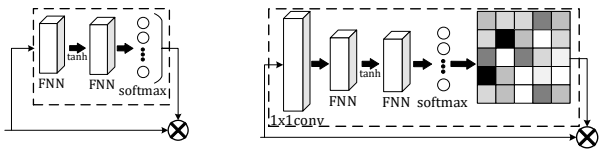
IV. Experiments

A. Baseline Approaches

To evaluate our work, we used two baseline approaches that will be referred to as baseline-1 and baseline-2.

Baseline-1 is inspired by potential field methods [12] and bearing-only navigation. We believe this is an appropriate comparison to our zero-range sensing architecture, as it receives the same input. In the case of continuous terrain, an IMU can be used for bearing-only navigation, as the direction of the gradient, θ_G, at each point can be calculated from roll and pitch values. We define $T(x, y)$ as the height in each cell of our generated terrain. At each time-step, an action is chosen according to the potential field given by

$$U(x, y, yaw) = \big(heading(yaw)\big)^2 + \alpha \cdot \|\nabla T(x, y)\| \cdot \big(\cos\big(4 \cdot (\theta_G - yaw)\big)\big). \quad (2)$$



(a) (b)

Figure 3. Attention modules architecture. (a) non-spatial input. (b) spatial input.

The first term, heading, guides the UGV to the goal position and reaches a minimum if the UGV is directed toward the goal location. The potential field induced by the first term is shown in Fig. 4a. Given a gradient vector norm, $\|\nabla T(x,y)\|$, and direction, θ_G, maximum pitch and roll values are measured when the UGV is oriented in parallel and in perpendicular to θ_G, respectively. Examples of these directions can be seen in Fig. 4b, marked as black arrows. The second term aims to avoid hazardous orientations by steering the UGV away from directions with high pitch and roll values. This behavior is achieved by minimizing the second term. The field induced by this potential function is shown in Fig. 4b as red arrows. In the case of a goal positioned on a hill or in a valley, this behavior allows the UGV to reach it by ascending or descending terrain with high gradients. It is essential to observe that Baseline-1 is explicitly given the potential function and follows the potential function gradient. However, zero-range sensing implicitly learns the underlying potential function and does not apply gradient descent on the actual terrain.

Baseline-2 evaluates an ego-graph [8], which is a set of candidate paths that are constructed from sequences of the UGV's available actions. Ego-graph depth is defined by the actions' sequences length. Each path is evaluated by summing the cost at each point along the path. The total cost of a path is represented as

$$cost(path) = \sum_{q \in path} heading\left(yaw_q\right) + \alpha \cdot \left\|\nabla T\left(x_q, y_q\right)\right\|. \quad (3)$$

After choosing a path that minimizes cost, we execute only the first action from this path and repeat the evaluation process. When navigating in unknown environment, this planning strategy allows to incrementally compute the trajectory over the course of the navigation, as mentioned in [34]. An ego-graph with a depth of 1 is an appropriate comparison to our immediate-range sensing, as it only captures the one-step look-ahead position that can be reached by executing a single action. Additionally, ego-graph with a depth greater than 1 is a suitable comparison to our local-range sensing architecture. An example of ego-graph with a depth value of 5 is shown in Fig. 5.

B. Results

In this section, we compare the performance of our approach against the presented baselines. The evaluation was made on 100 pairs of randomly selected start and goal positions. Table I presents a success rate and average planning time comparison between approaches. The reported average planning time is calculated over several successful plans and presented in seconds. DRL methods show significant improvement as compared to their counterparts. Our experiments demonstrate that planning times for low dimension inputs are similar for DRL planners when compared to their counterparts, baseline-1 for zero-range sensing, and baseline-2 depth-1 for immediate-range sensing. However, as expected when dealing with high dimensional input, our DRL approach provides better planning in under 2 seconds, which is considerably less time than baseline-2 depth-5, which requires an average of 130 seconds per path.

Fig. 6 shows prototypical examples of paths generated by the baselines compared to our proposed approach. The only

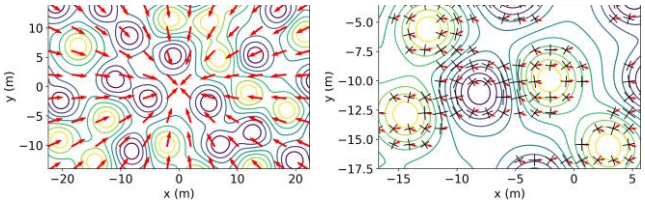


Figure 4. Illustration of forces produced by baseline-1 potential function. a) Attraction forces to goal position. b) Repulsive forces (red arrow) from orientations with high roll or high pitch angles.

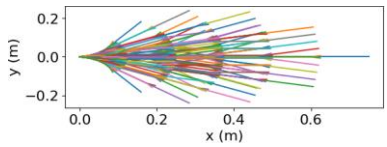


Figure 5. Ego graph of depth 5 spanned by available actions - forward, right, and left.

available input for our navigation problem regarding the goal position is the direction toward the goal. This constraint creates a situation where it is crucial that hills or valleys are not avoided, in case the goal position is located on a hill or in a valley.

Therefore, in Fig. 6a, baseline-1 and DRL zero-range sensing successfully plan a path toward the goal while ascending and descending hills in order to reach the goal position. However, as seen in Fig. 6b, baseline-1 fails to reach the goal, while DRL zero-range sensing is inclined to traverse along contours, equi-elevation lines, which helps it avoid high gradient areas if scaling them is not necessary. DRL zero-range sensing also exhibits more robust maneuvers in order to traverse the terrain and reach the goal safely.

In Fig. 6c, DRL zero-range sensing is unable to react quickly enough to the sudden change in elevation, whereas, DRL immediate-range sensing and baseline-2 depth-1 can sense the imminent danger and avoid it. Fig. 6d shows how the DRL immediate-range sensing exhibits better maneuvers and terrain understanding compared to its counterpart, baseline-2 depth-1.

Some areas of the terrain pose a more significant challenge for DRL immediate-range sensing due to a broad area of high gradients, as can be seen in Fig. 6e. However, DRL local-range sensing which senses elevation in a 3 m box which around the robot, and baseline-2 depth-5, can induce a non-myopic policy which presents long-term planning, to circumvent hazardous areas while still moving toward the goal. Fig. 6f presents how DRL local-range sensing can overcome a challenging terrain configuration by bypassing a steep hill and understanding that it must execute

TABLE I. PERFORMANCE OVER TEST PATHS

Approach	*Success (%)*	*Avg Planning Time (sec)*
Baseline-1	26	0.15
DRL Zero-range Sensing	**48**	0.24
Baseline-2 Depth-1	61	0.73
DRL Immediate Range Sensing	**69**	0.34
Baseline-2 Depth-5	69	130
DRL Local Range (3m box centered)	**77**	1.51
DRL Local Range (3m box ahead)	**82**	1.72

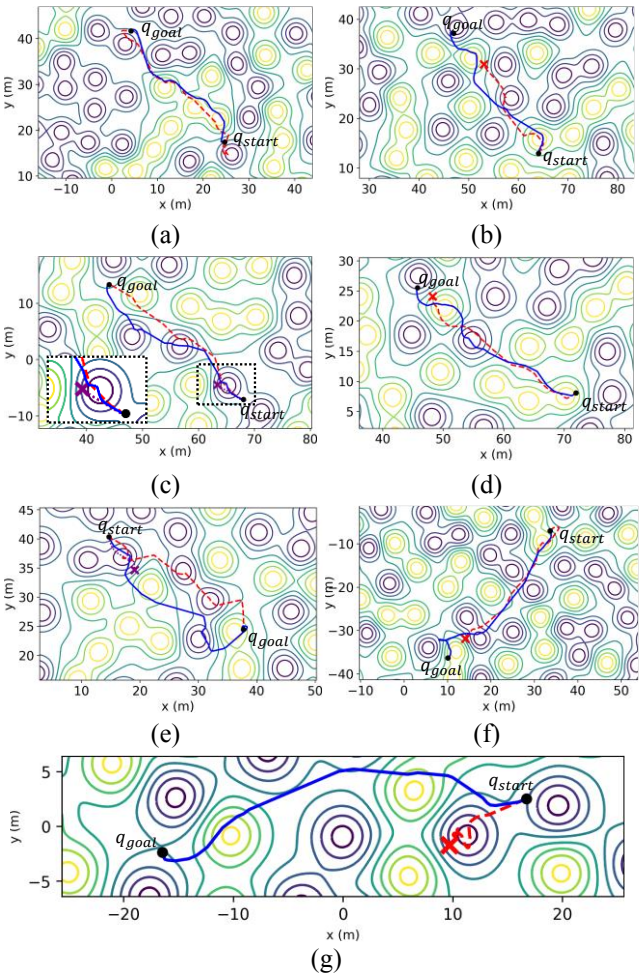


Figure 6. Paths examples comparing different baselines with suitable input types. (a) zero range sensing (solid blue) and baseline_1(dashed red). (b) zero range sensing (solid blue) and baseline_1(dashed red). (c) immediate range sensing (solid blue), baseline_2 with depth value of 1 (dashed red) and zero range sensing (dotted purple). (d) immediate range sensing (solid blue), baseline_2 with depth value of 1 (dashed red). (e) local sensing 3m box centered around the robot (solid blue), baseline 2 with depth value of 5 (dashed red) and immediate range sensing (dotted purple). (f) local sensing 3m box centered around the robot (solid blue) and baseline 2 with depth value of 5 (dashed red). (g) local sensing 3m box ahead of the robot (solid blue) and local sensing 3m box centered around the robot (dashed red).

a difficult maneuver to reach the goal, in contrast to baseline-2 depth-5 which fails.

Lastly, Fig. 6g demonstrates how DRL local-range sensing that senses elevation in a 3 m box that lies in front of the UGV can produce a superior non-myopic policy compared to sensing the elevation in 3 m box that is centered around the UGV. Sensing 3 m ahead with combination of DRL allowed the UGV to avoid the "high cliff trap", whereas more nearsighted sensing resulted in a failed attempt to steer away from the cliff.

Of the 18% failures in row 7 of Table I, there are two main failure modes. The first failure mode (13% of all trials) occurs in local minimum areas of the potential field which translates to broad areas of high gradients in the terrain. This is a fundamental issue with local planners. This issue can be addressed by increasing the sensing range (as was exemplified in better success rates in Table I), injecting sequences of paths generated by a global planner, or training the network with long replay sequences.

The second, less common mode (5% of all trials), includes pathological starting poses due to the randomness of starting poses. For example, starting on a high gradient surface with the vehicle's orientation opposite to the target direction. It is unlikely that during training the vehicle experiences such states which can lead to a poor performance under these extreme situations. This issue can be alleviated by changing the starting poses' distribution during training.

C. Attention Visualization

The inputs' importance can be analyzed through extracting the softmax outputs from the self-attention modules. In Fig. 7 we highlight the terrain area which is most attended to in the spatial input. The attention over the height map is focused on the front-left and front-right areas relative to the UGV. It can be seen that the UGV's attention is focused on a nearby high gradient hazardous area. The focused attention mainly occurs in states where the UGV is located on non-threatening gradients. The attention shifts as the UGV proceeds. For non-spatial inputs, we observed that when the angle between the UGV and the target increases, the attention to it increases as well. Similarly, when facing high roll values, the UGV has an increased attention to the roll input versus the pitch input, and vice versa. This attention behavior of the non-spatial inputs remains the same across the examined input types. Our analysis implies that the attention over the height map is mainly used for searching for areas with small gradients and not for ascending or descending maneuvers, as opposed to roll and pitch attention which increases when the UGV is located on high gradients.

V. Dynamic Validation

In real terrain navigation, one must consider the vehicle's dynamics and the interaction between the vehicle and the terrain, that results in sliding and wheel slip. However, modeling dynamic constraints can be hard to construct and tune [35], let alone incorporate dynamics into planning, which requires expert knowledge. We used our DRL method, on terrain modelled in Gazebo [36] with a simulated Clearpath Robotics Jackal as the UGV. Our algorithm was able to capture the vehicle's dynamics and interaction with the terrain and plan accordingly.

A. Terrain Modification

Our main objective was to validate that our approach can capture vehicle dynamics and interaction with the terrain, while performing maneuvers to traverse the continuous terrain and avoid binary obstacles.

An example of a path on the generated terrain used in the dynamic validation was shown in Fig. 1. Height map of the training terrain can be seen in Fig. 8, where brighter areas

Figure 7. Attention visualized in the kinematic simulation. The most attended area (highlighted) is focused on nearby high gradients. The arrow indicates the orientation of the UGV.

indicate a higher surface. The UGV's start position is located on top of the left-most hill, while the goal position is located on top of the right-most hill. This configuration forces the UGV to climb down the left hill, circumnavigate the circle-shaped binary obstacles that protrude from the ground, cross a ditch, circumnavigate rectangular holes in the terrain, and climb the right hill to reach the goal position.

B. Architecture Extension and Training

Low friction environments pose a challenge to planetary mobile robots as high wheel slip and sliding motions occur. Approximations to the wheel slip and sliding model are often used in order to compensate for such motions [37]. To implicitly learn how to compensate for wheel slip and sliding motions, without explicitly providing a model approximation, we extend our architecture's input and provide the geometric transformation between two successive timesteps and the corresponding action. The geometric transformation is represented as the difference in the x and y position in the UGV's local frame. The action is encoded as a one-hot vector.

Training was done by selecting a different start position for each episode, which enabled exploration of different behaviors and areas. A friction value was randomly selected out a range of feasible values at the beginning of each episode. We assumed the UGV has local range sensing and thereby used the DRL Local Range (3m box centered) architecture. Due to the locality of our range sensing, we argue that planning to traverse the full path can be done by learning to traverse segments of the full path and this assumption is validated in our experiment.

C. Results

Our DRL method was able to successfully navigate in the training terrain, whereas the kinematic baseline-2 depth 5 planner failed in dynamically challenging areas with increased slip, which is evident in Fig. 8. To test the generalization ability of our navigation method, a test terrain was generated. The test terrain, shown in Fig. 9, introduces binary obstacles with different sizes and angles, and requires to encounter the obstacles in a different sequence than in the training terrain. While never training on the test terrain, our trained DRL planner navigated successfully from desired start positions to goal positions, as can be seen in Fig. 9.

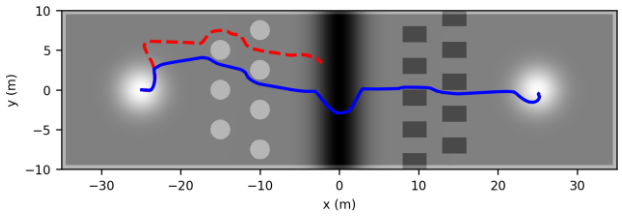


Figure 8. Our DRL method (solid blue) successfully navigating through the dynamic training terrain while baseline-2 depth-5 (dashed red) fails.

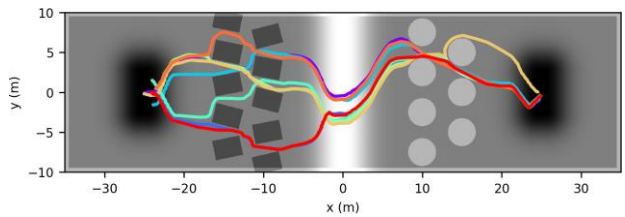


Figure 9. Paths examples generated by our DRL method in the dynamic test terrain.

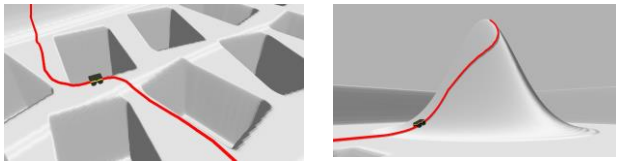


Figure 10. DRL method exhibiting different maneuvers for overcoming a continuous obstacle and binary obstacles.

TABLE II. PERFORMANCE UNDER DIFFERENT FRICTION SETTINGS

	Approach	*Original Architecture*	*Extended Architecture*
High friction setting	***Success***	10/10	10/10
	Avg Timesteps	397.5	**381.1**
Medium friction setting	***Success***	4/10	**10/10**
	Avg Timesteps	1960.5	**628.8**
Low friction setting	***Success***	0/10	**10/10**
	Avg Timesteps	-	**1046.7**

(a) (b)

Figure 11. Attention visualized in the dynamic simulation. The most attended area (highlighted) is focused on a nearby hole (purple).

Additionally, Fig. 10 shows how different types of obstacles are handled with suitable maneuvers in order to reach the goal. In order to climb the hill Fig. 10 (right), the wheel-terrain interaction and vehicle dynamics need to be taken into account, and a spiral trajectory is generated to avoid flipping due to high pitch angles. For the purpose of circumnavigating the hole Fig. 10 (left), it is crucial to stay clear of the hole perimeter, or the UGV might fall down. See supplementary video for dynamic simulation examples.

We compare the performance of our extended architecture versus the original architecture in the hill climbing task under different friction settings. Table II presents the number of successful trials and average timesteps required to climb the hill. Our extended architecture is able to handle different levels of friction and climb the hill in less timesteps than the original architecture.

D. Attention Visualization

The dynamic simulation setting introduced discrete obstacles into the planning problem. Similar to visualizing the attention in the kinematic simulation, we show that the attention module is also attending to hazardous areas which should be avoided in the dynamic simulation. As shown in Fig. 11 the UGV's attention is focused on the nearby hole in order to adjust its position to avoid falling. Again, the attention shifts as the UGV proceeds.

VI. CONCLUSION

In this work we presented a DRL based method to locally navigate unknown rough terrain with zero-range to local-range sensing. We showed that our method improved planning time and planning success rate compared to traditional local planning methods. Our method presents

reward shaping, which provides a dense reward signal in a goal-directed task. Also, we extended and validated our approach on a dynamic simulation and showed that our method successfully managed to capture the vehicle's dynamic and interaction with the terrain while traversing continuous terrain with adjustable degree of friction, avoiding a variety of discrete obstacles, and navigating safely from start position to goal position. The attention mechanism unprecedented applied to rough terrain navigation problem provided insights regarding the learnt policy. Future work will include incorporating expert knowledge presented by a global planner or a human operator into the learning process, and applying a *sim-to-real* transfer to a physical UGV.

REFERENCES

[1] H. M. Choset *et al.*, *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[2] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag.*, vol. 4, no. 1, pp. 23–33, 1997.

[3] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *Dep. Pap.*, p. 323, 1992.

[4] I. Ulrich and J. Borenstein, "VFH+: Reliable obstacle avoidance for fast mobile robots," in *Proceedings. 1998 IEEE international conference on robotics and automation (Cat. No. 98CH36146)*, 1998, vol. 2, pp. 1572–1577.

[5] T. Krajn'ik, J. Faigl, V. Vonásek, K. Košnar, M. Kulich, and L. Pvreučil, "Simple yet stable bearing-only navigation," *J. F. Robot.*, vol. 27, no. 5, pp. 511–533, 2010.

[6] S. L. Laubach, J. Burdick, and L. Matthies, "An autonomous path planner implemented on the Rocky 7 prototype microrover," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, 1998, vol. 1, pp. 292–297.

[7] A. Ettlin and H. Bleuler, "Rough-terrain robot motion planning based on obstacleness," in *2006 9th International Conference on Control, Automation, Robotics and Vision*, 2006, pp. 1–6.

[8] A. Lacaze, Y. Moscovitz, N. DeClaris, and K. Murphy, "Path planning for autonomous vehicles driving over rough terrain," in *Proceedings of the 1998 IEEE International Symposium on Intelligent Control (ISIC) held jointly with IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA) Intell*, 1998, pp. 50–55.

[9] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *J. F. Robot.*, vol. 26, no. 3, pp. 308–333, 2009.

[10] Y. Yi, M. Fu, Z. Hao, G. Xiong, and C. Sun, "Control methods of mobile robot rough-terrain trajectory tracking," *2010 8th IEEE Int. Conf. Control Autom. ICCA 2010*, pp. 731–738, 2010.

[11] S. Shimoda, Y. Kuroda, and K. Iagnemma, "Potential field navigation of high speed unmanned ground vehicles on uneven terrain," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 2828–2833.

[12] A. Ettlin, P. Büchler, H. Bleuler, and others, "Rough-terrain robot motion planning based on topology and terrain constitution," in *18th International Congress of Mechanical Engineering*, 2005.

[13] K. Iagnemma, S. Shimoda, and Z. Shiller, "Near-optimal navigation of high speed mobile robots on uneven terrain," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 4098–4103.

[14] P. Ogren and N. E. Leonard, "A convergent dynamic window approach to obstacle avoidance," *IEEE Trans. Robot.*, vol. 21, no. 2, pp. 188–195, 2005.

[15] A. Tahirovic and G. Magnani, "An extension to rough terrains of the MPC/CLF mobile vehicle navigation approach," in *Proceedings of the 10th International Conference on Mobile Robots and Competitions (ROBTICA 2010)*, 2010, vol. 10, p. 12.

[16] C. Ye and J. Borenstein, "A method for mobile robot navigation on rough terrain," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, 2004, vol. 4, pp. 3863–3869.

[17] A. H. Qureshi, A. Simeonov, M. J. Bency, and M. C. Yip, "Motion planning networks," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2118–2124.

[18] T. Jurgenson and A. Tamar, "Harnessing Reinforcement Learning for Neural Motion Planning," *arXiv Prepr. arXiv1906.00214*, 2019.

[19] K. Zhang, F. Niroui, M. Ficocelli, and G. Nejat, "Robot navigation of environments with unknown rough terrain using deep reinforcement learning," in *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2018, pp. 1–7.

[20] I. Sorokin, A. Seleznev, M. Pavlov, A. Fedorov, and A. Ignateva, "Deep attention recurrent Q-network," *arXiv Prepr. arXiv1512.01693*, 2015.

[21] A. Manchin, E. Abbasnejad, and A. van den Hengel, "Reinforcement learning with attention that works: A self-supervised approach," in *International Conference on Neural Information Processing*, 2019, pp. 223–230.

[22] M. Hessel *et al.*, "Rainbow: Combining improvements in deep reinforcement learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[23] D. Coombs, K. Murphy, A. Lacaze, and S. Legowik, "Driving autonomously off-road up to 35 km/h," in *Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No. 00TH8511)*, 2000, pp. 186–191.

[24] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[25] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[26] A. Vaswani *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[27] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv Prepr. arXiv1508.04025*, 2015.

[28] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine, "Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1–8.

[29] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 AAAI Fall Symposium Series*, 2015.

[30] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," *arXiv Prepr. arXiv1511.06581*, 2015.

[31] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp. 449–458.

[32] A. A. Taiga, W. Fedus, M. C. Machado, A. Courville, and M. G. Bellemare, "Benchmarking Bonus-Based Exploration Methods on the Arcade Learning Environment," *arXiv Prepr. arXiv1908.02388*, 2019.

[33] J. Randløv and P. Alstrøm, "Learning to Drive a Bicycle Using Reinforcement Learning and Shaping.," in *ICML*, 1998, vol. 98, pp. 463–471.

[34] T. Schouwenaars, J. How, and E. Feron, "Receding horizon path planning with implicit safety guarantees," in *Proceedings of the 2004 American control conference*, 2004, vol. 6, pp. 5576–5581.

[35] C. Wellington and A. Stentz, "Online adaptive rough-terrain navigation vegetation," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, 2004, vol. 1, pp. 96–101.

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

[37] T. M. Howard and A. Kelly, "Optimal rough terrain trajectory generation for wheeled mobile robots," *Int. J. Rob. Res.*, vol. 26, no. 2, pp. 141–166, 2007.