

This work was written as part of one of the author's official duties as an Employee of the United States Government and is therefore a work of the United States Government. In accordance with 17 U.S.C. 105, no copyright protection is available for such works under U.S. Law.

Public Domain Mark 1.0

<https://creativecommons.org/publicdomain/mark/1.0/>

Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing scholarworks-group@umbc.edu and telling us what having access to this work means to you and why it's important to you. Thank you.

CoverNav: Cover Following Navigation Planning in Unstructured Outdoor Environment with Deep Reinforcement Learning

Jumman Hossain¹, Abu-Zaher Faridee¹, Nirmalya Roy¹, Anjan Basak², Derrik E. Asher²

¹Department of Information Systems, University of Maryland, Baltimore County, USA

²DEVCOM Army Research Lab, USA

¹{jumman.hossain, faridee1, nroy}@umbc.edu

²anjon.sunny@gmail.com, ²derrik.e.asher.civ@army.mil

Abstract—Autonomous navigation in off-road environments has been extensively studied in the robotics field. However, navigation in covert situations where an autonomous vehicle needs to remain hidden from outside observers remains an under-explored area. In this paper, we propose *CoverNav*, a novel Deep Reinforcement Learning (DRL) based algorithm, for identifying covert and navigable trajectories with minimal cost in off-road terrains and jungle environments in the presence of observers. *CoverNav* focuses on unmanned ground vehicles seeking shelters and taking covers while safely navigating to a predefined destination. Our proposed DRL method computes a local cost map that helps distinguish which path will grant the maximal covertness while maintaining a low-cost trajectory using an elevation map generated from 3D point cloud data, the robot's pose, and directed goal information. If an observer is spotted, *CoverNav* enables the robot to select natural obstacles (e.g., rocks, houses, trees, etc.) and use them as shelters to hide behind. We evaluate *CoverNav* using the Unity simulation environment and show that it guarantees dynamically feasible velocities in the terrain when fed with an elevation map generated by another DRL-based navigation algorithm. Additionally, we evaluate *CoverNav*'s effectiveness in achieving a maximum goal distance of 12 meters and its success rate in different elevation scenarios with and without cover objects. We observe competitive performance comparable to state-of-the-art (SOTA) methods without compromising accuracy.

Index Terms—deep reinforcement learning, simulated terrain environment, off-road path planning, cover detection, and semantic segmentation.

I. INTRODUCTION

Autonomous ground vehicles, particularly, have great potential as companions to humans, seamlessly collaborating with or actively participating in tasks. They can strategically position themselves in advantageous locations to optimize operations and achieve objectives more efficiently. For autonomous ground vehicles to be effective in various scenarios, they must be able to navigate through complex terrains. A significant aspect of this navigation challenge is the concept of covert path planning, which involves unmanned ground vehicles (UGVs) utilizing physical features of the terrain to minimize visibility, enhancing their ability to remain concealed and reducing detection risk. This is especially important in situations where it's necessary to move the robot while remaining hidden from observers. Al Marzouqi and Jarvis [3] conducted one of the earliest comprehensive surveys on robotic covert path planning, outlining the various strategies and techniques used in the field. Subsequently, Ninomiya et al. [4] presented a planning framework that satisfies multiple spatial constraints imposed on the path, such as staying

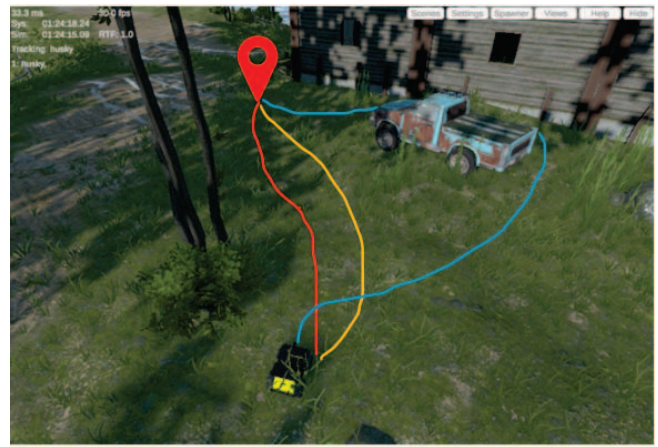


Fig. 1: During navigation in unstructured outdoor environments, our proposed *CoverNav* algorithm directs the Clearpath Husky robot towards nearby obstacles that provide the highest cover. In simulation, the robot detects the position of obstacles in its vicinity and moves towards them to find cover. If there are multiple obstacles in the robot's field of view, *CoverNav* prioritizes the obstacle with the highest cover. In the accompanying image, we show the robot navigating using our DRL-based method in blue, and state-of-the-art navigation planning algorithms, TERP [1] in red and GA-Nav [2] in yellow. In the image, the robot initially detects a bush as its cover, moves towards it, and then changes direction when it no longer perceives any cover in that direction. It quickly finds shelter between a truck and a cottage until it reaches its destination.

behind a building, walking along walls, or avoiding the line of sight of patrolling agents. As evident from these works, to achieve effective covert path planning, UGVs must have an integrated navigation planning system that allows them to adapt their movements based on the environment and select routes that optimize cover and concealment. UGVs should adapt their movements based on the environment, selecting routes that optimize cover and concealment, anticipating potential observers. Developing such a method faces challenges in distinguishing safe regions while considering robot properties and terrain cover. Deep Reinforcement Learning (DRL) is a promising approach in addressing these limitations [5], [6], enabling UGVs to make strategic decisions based on the environment and maximize a reward function. DRL techniques have excelled in various

navigation tasks, including detecting unsafe elevation changes, efficient trajectory planning, and guiding UGVs around moving objects [7]. However, they often do not account for the crucial aspect of minimizing the likelihood of being seen by an outside observer. Incorporating these strategic considerations into the DRL framework is an essential step toward creating more effective and practical UGVs for off-road navigation.

In this work, we present a novel Deep Reinforcement Learning (DRL) based method for identifying covert and navigable trajectories in off-road terrains and jungle environments. Our approach focuses on seeking shelters and covers while safely navigating to a predefined destination with the objective of minimizing the visibility of the robot to potential observers. In this context, “cover” refers to physical objects that conceal the robot from detection by observers, such as buildings, trees, hills, vehicles, and other obstacles that obstruct the robot’s view or hide it from an observer’s line of sight. Our main contributions are summarized as the following:

- **Designing a Novel Reward Function :** Our proposed DRL network utilizes a novel reward function to effectively learn optimal off-road terrain features with covers in simulated environments. By rewarding the agent for moving towards the objective, penalizing for deviating from it, promoting low altitude, encouraging proximity to cover, and maintaining stability, our approach is able to learn quickly and efficiently. This results in the ability to plan reliable and steady trajectories while maintaining a safe distance from obstacles in order to seek out covers.
- **Designing a Novel Cover Detection Algorithm :** We present a novel cover detection algorithm that leverages state-of-the-art semantic segmentation techniques to precisely identify and locate cover objects in unstructured outdoor environments. By exploiting the rich semantic information provided by segmentation, our algorithm can accurately identify and localize cover objects, providing the necessary information to the robot to take appropriate actions for seeking shelter when needed. The proposed approach thus enhances the safety and robustness of the robot navigation system in challenging outdoor environments.
- **Integrated Cover-Following Navigation Planning Approach:** We propose an integrated approach that combines Terrain Elevation-based Reliable Path Planning (TERP) [1] and Dynamic Window Approach-based Reinforcement Learning (DWA-RL) [8] with our cover detection algorithm to enhance the navigation of a robot in an unstructured outdoor environment. The TERP method provides reliable and stable path planning for the robot, while the DWA-RL approach facilitates navigation in an environment with mobile obstacles. The integration of these methods enables the robot to locate cover and shelter, navigate safely (i.e., maintain a stable trajectory and move without being detected by potential threats) to a predetermined destination, and avoid dynamic obstacles. This combined approach provides a more robust and efficient navigation strategy for the robot in complex outdoor environments.
- **Experimenting in Various Simulated Unstructured Environments:** In order to evaluate the effectiveness of our proposed approach, we conducted extensive simulations

using a Clearpath Husky robot in various unstructured environments, such as off-road terrains and jungles with varying levels of vegetation density. CoverNav exhibits competitive performance in success rate and trajectory length, comparable to state-of-the-art (SOTA) methods while maintaining high accuracy. We also tested our strategy in a realistic forest environment with diverse terrain features, extensive areas of cover, and different elevations. Our experiments demonstrate that *our approach can successfully achieve the highest possible goal distance of 12 meters and high success rates* (i.e., the percentage of trials in which the robot successfully completes the specified goal distance) in navigating through terrains with varying degrees of elevation and cover. These results provide valuable insights into the potential real-world performance of our approach.

For a more detailed version of our work, we direct the reader to [9].

II. RELATED WORK

In this section, we discuss existing work on robot navigation in unstructured outdoor environments.

A. Semantic Segmentation for Off-Road Surface Traversability

In many scenarios, a robot encounters unstructured outdoor terrains with different levels of navigability. Several recent works have addressed these challenges of off-road navigation by proposing novel methods with semantic segmentation [10], [11]. GA-Nav [2] is a learning-based semantic segmentation method that identifies different terrain groups in off-road and unstructured outdoor terrains. It balances segmentation accuracy and computational overhead and outperforms state-of-the-art segmentation methods by using coarse-grained segmentation. For instance, it recognizes trees and poles simply as obstacles instead of categorizing them as individual classes. In the context of our work, we utilize the potential of the semantic segmentation method to effectively detect and classify cover objects in the environment.

B. Deep Reinforcement Learning Based Off-road Navigation

In recent studies, Deep Reinforcement Learning (DRL) has been utilized to improve navigation strategies in complex, off-road environments. Taking the challenge of uncertainty, Zhang et al. [5] demonstrated the potential of DRL for navigating environments with unknown rough terrain. Wiberg et al. [12] proposed a DRL approach for vehicle control in rough terrains, concentrating on developing an effective mechanism to handle complex and uneven surfaces. Similarly, Josef and Degani [13] presented a DRL method for safe local planning of a ground vehicle in unknown rough terrain, underscoring the necessity for robust navigation in the face of unpredictability and hazards associated with rough terrains. TERP [1] is a recent DRL-based method specialized in robust navigation in uneven outdoor terrains. It effectively identifies unsafe regions with high elevation gradients and calculates local least-cost waypoints to avoid untraversable areas.

Existing techniques in autonomous navigation have made significant progress, but they often overlook the vital aspect of finding hiding or cover spots in outdoor terrains. This feature becomes critical in scenarios where concealing the UGVs from potential observers is of utmost importance. In our method, we introduce a novel approach that integrates cover-seeking into DRL-based navigation for UGVs in off-road terrains.

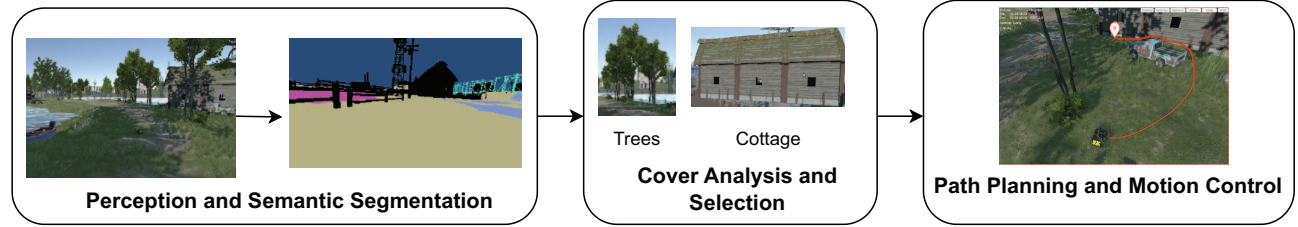


Fig. 2: Overview of the CoverNav Navigation Planning System: The process begins with *Perception and Semantic Segmentation*, where the environment is sensed and objects are classified (block 1). This feeds into the *Cover Analysis and Selection* module (block 2), which identifies potential cover objects and selects the optimal one. The selected cover object is then used by the *Path Planning and Motion Control* modules (block 3) to both generate a safe and optimal trajectory towards the object and executes this trajectory through the robot's physical movements.

III. METHODOLOGY

In this section, we present an overview of the primary modules of *CoverNav*, progressively providing specific details of our contributions and where they fit in.

A. Overview of CoverNav Navigation Planning

The *CoverNav* navigation planning system (Figure 2) comprises three interdependent modules. The details of the modules are articulated below:

1) Perception and Semantic Segmentation

This module receives and processes image and point cloud sensor data from the environment. We first infer the robot's pose, twist, and elevation map from this data. Semantic segmentation techniques are applied to raw sensor data to detect and classify objects in the environment.

2) Cover Analysis and Selection

As the semantic segmentation data is passed to this module, we utilize our novel cover detection Algorithm (described later Section III-B2) to identify potential cover providing natural obstacles such as trees, rocks, bushes, etc. This process takes into account the distance of objects from the robot, as well as the potential for collisions and the amount of cover provided. Once potential cover objects are identified, this module calculates the 3D Euclidean distance between the robot and each potential cover object. It then applies predefined selection criteria, including factors such as distance, and visibility, to determine the most suitable cover object.

3) Path Planning and Motion Control

This module integrates the path planning and motion control functionalities to facilitate effective navigation of the robot. In this module, we combine TERP [1] (a Deep Deterministic Policy Gradient (DDPG) [14] algorithm) with our novel reward function to generate a trajectory that guides the robot towards the selected cover object, while avoiding obstacles. The computed trajectory takes into consideration the robot's current position, the cover object's location, and the surrounding environment. The goal is to determine a safe and optimal trajectory that maximizes cover, minimizes visibility, and assures obstacle avoidance. This trajectory is dynamically updated in response to changes in the environment or the detection people within the environment. The motion control aspect utilizes the DWA-RL [8] algorithm, which combines the strengths of the Dynamic Window Approach (DWA) [15] and Deep Reinforcement Learning (DRL) methods. It determines dynamically feasible

velocities that enable the robot to adhere to the trajectory while ensuring collision avoidance and maintaining stability.

These three modules work together to enable the robot to autonomously and intelligently navigate towards its goal, maximizing safety, and minimizing the risk of detection. The system architecture is shown in Fig. 3, where the inputs from the sensors are processed through different modules to produce the output of dynamically feasible velocities for the robot to follow.

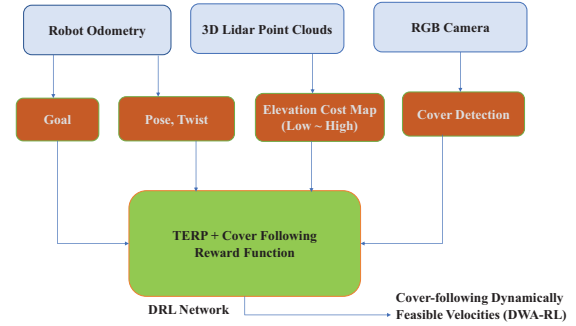


Fig. 3: Overview of CoverNav System Architecture.

B. Problem Formulation

We formulate our problem as a Markov decision process (MDP). Our MDP consists of a tuple (S, A, P, R, γ) where

- S is the state space representing the robot's state. It includes the robot's position and orientation, as well as the presence and location of obstacles and covers in the environment.
- A is the action space representing the robot's actions. It includes the robot's movement in different directions and its rotation.
- P is the transition function representing the probability of moving from one state to another after taking an action.
- R is the reward function representing the immediate reward the robot receives after taking an action.
- γ is the discount factor representing the importance of future rewards.

Our objective is to find a policy $\pi(s)$ that maps a state s to action a , such that the expected cumulative reward of the robot is maximized.

1) Reward Function Design

Our reward function is designed to encourage desirable robot behaviors and discourage undesirable ones. It includes five components:

r_{goal} : This term measures progress towards the goal by subtracting the current distance from the previous distance. A positive reward is assigned if the robot moves closer to the goal, indicating progress, while a negative reward is assigned if the robot moves farther away.

$$r_{goal} = d_{t-1} - d_t \quad (1)$$

r_{dir} : This reward component encourages the robot to maintain a consistent heading toward the goal. A smaller difference indicates a more consistent heading, resulting in a higher reward.

$$r_{dir} = -|\theta_t - \theta_{t-1}| \quad (2)$$

r_{stab} : This reward component evaluates the stability of the robot based on its roll and pitch angles (ψ_t). The reward is calculated using an exponential function, where the stability reward decreases as the roll and pitch angles deviate from zero. Smaller values of ψ_t result in higher stability rewards.

$$r_{stab} = \exp(-\psi_t^2) \quad (3)$$

r_{elev} : This term encourages the robot to move towards areas of higher elevation by penalizing deviations from the target elevation. Mathematically, it can be represented as:

$$r_{elev} = \sum_{i=1}^n w_{elev} \Delta h_i \quad (4)$$

Δh_i is calculated as the difference between the elevation at the current position and the i^{th} previous position, given by:

$$\Delta h_i = h_{cur} - h_i \quad (5)$$

r_{cover} : This term encourages the robot to seek cover objects to hide behind or beneath. It is computed as follows:

$$r_{cover} = \begin{cases} 0, & \text{if } d_{coverObject} > 1.5w_{min} \\ d_{coverObject} - 0.5w_{min}, & \text{if } 0.5w_{min} \leq d_{coverObject} \leq 1.5w_{min} \\ -1000, & \text{if } d_{coverObject} < 0.5w_{min} \end{cases} \quad (6)$$

It is calculated based on the distance to the nearest cover object, represented by $d_{coverObject}$, and the shortest external dimension of the robot, denoted as w_{min} .

2) Cover Detection

To accurately locate and label potential cover objects, we utilize the ARL Unity Framework's semantic segmentation, which assigns class labels to each pixel in the image. We focus on objects like buildings, trees, and rocks that offer protection and concealment. Calculating the 3D distance between the robot and these cover objects using the Euclidean formula helps the robot make informed navigation decisions. The distance calculation is represented by Equation (7):

$$d_{coverObject} = \sqrt{(x_{robot} - x_{cover})^2 + (y_{robot} - y_{cover})^2 + (z_{robot} - z_{cover})^2} \quad (7)$$

where $d_{coverObject}$ represents the distance between the robot and the cover object, and $(x_{robot}, y_{robot}, z_{robot})$ and $(x_{cover}, y_{cover}, z_{cover})$ represent the 3D coordinates of the robot and the cover object, respectively. Algorithm 1 then determines whether the robot is currently in cover by considering the detected objects' classes and detection confidence. A confidence threshold of 0.85 ensures only objects with high detection confidence are considered. The cover detection algorithm efficiently iterates through detected objects (time complexity: $O(N)$) and determines cover based on specific thresholds, ensuring effective object selection without compromising its computational efficiency.

Algorithm 1: Algorithm for Detecting Cover

Input: $objC$ ▷ Object Classes
Input: $objL$ ▷ Object Locations
Input: rL ▷ Robot Location
Input: cf ▷ Confidence Values
Output: $isCover$ ▷ Cover or not cover

```

1 Function DetectCover ( $objC, objL, rL, cf$ ) :
2    $coverDistance \leftarrow \infty$ ;
3    $isCover \leftarrow False$ 
4    $N \leftarrow length(objC)$ 
5   foreach  $i \in N$  do
6     if
7        $objC_i \in \{trees, bushes, rocks, cottage, building, houses$ 
8        $disabled\ vehicles\}$  and  $cf_i \geq 0.85$  then
9          $dist_i \leftarrow ||objL_i - rL||$ 
10        if  $dist_i < coverDistance$  then
11           $coverDistance \leftarrow dist_i$ 
12        end
13      end
14    if  $coverDistance \leq 10$  then
15       $isCover \leftarrow True$ 
16    end
17  return  $isCover$ ;

```

3) Dynamic Window Approach with Deep Reinforcement Learning

To enhance navigation in dynamic environments with mobile obstacles, we employ a recent approach [8] that combines the benefits of Dynamic Window Approach (DWA) [15] and Deep Reinforcement Learning (DRL). Unlike the limitations of DWA in reacting to changes in the environment, this method incorporates a reward function that enhances the robot's spatial awareness of moving obstacles. It encourages the robot to navigate away from obstacle headings, facilitating maneuvers around dynamic obstacles. Integrating DRL with the high-level path planner reduces the complexity and accelerates the training process. This formulation maintains dynamic feasibility guarantees while incorporating time-dependent changes in the environment, leading to a significantly reduced dimension of the observation space compared to other DRL methods.

C. Training Strategy

Our training strategy combines exploration via a DRL agent with the exploitation of DWA's dynamic feasibility guarantees. The DRL agent initially explores the environment using random actions to learn the optimal policy. After sufficient episodes, the DRL agent is tested with DWA to ensure obstacle avoidance. The training parameters include a learning rate of 10^{-4} for both actor and critic networks, a batch size of 64, and a replay buffer size of 100,000. We use the Adam optimizer with decay rates of 0.9 and 0.999 for the first and second moments, respectively. The discount factor γ is set to 0.99, and the exploration noise scale σ is set to 0.1. The training process consists of 100 episodes, with each lasting for 100 time-steps. The robot's initial position is randomized within the start zone at the beginning of each episode. Training is terminated at episode 100

as further iterations did not significantly improve the success rate and trajectory length, given computational constraints.

IV. EXPERIMENTS AND RESULTS

We explain our implementation and experimental details using a Husky robot in the ARL Unity simulation environment.

A. Implementation Details

The model is tested in the high-fidelity Unity simulator, a realistic outdoor environment with diverse terrain and objects. Our DRL network is implemented in PyTorch and trained using simulated uneven terrains with a Clearpath Husky robot in ROS Melodic and Unity Simulation framework. The simulated Husky robot is mounted with a Velodyne VLP16 3D LiDAR. The network is trained in a workstation with a 10th Generation Intel Core i9-10850K processor and an NVIDIA GeForce RTX 3090 GPU.

We utilize the ARL Unity semantic segmentation framework and apply custom labeling to ignore small objects that do not provide cover for the robot. We use the Octomap [16] and Grid-map [17] ROS packages to obtain the elevation map using the point clouds from the Velodyne ROS package. Our network's computational load is low enough, allowing it to operate in real-time on a standard laptop with ROS tools.

B. Baselines

We compare the performance of our algorithm against the following navigation strategies:

- **DWA [15]:** A popular algorithm that focuses on real-time obstacle avoidance and trajectory planning. It calculates feasible velocities based on the current robot state and dynamically selects the best velocity combination to avoid collisions and reach the goal.
- **GA-Nav [2]:** A learning-based semantic segmentation method that aims to identify different terrain groups in unstructured and outdoor environments. It has shown promising results in accurately classifying terrain surfaces and improving navigation performance in complex terrains.
- **TERP [1]:** A DRL-based planner specifically designed for navigating uneven outdoor terrains. It incorporates terrain elevation information to identify and avoid unsafe regions by computing local least-cost waypoints.

C. Evaluation Metrics

Due to fundamental differences between *CoverNav* and some baselines (e.g., neither DWA and GA-Nav are DRL-based methods), direct reward score comparison is not meaningful. Instead, we assess navigation performance using the following metrics for a fair comparison.

Success Rate: The ratio (in percentage) between the number of times the robot successfully accomplished its objective and the number of total tries, while avoiding obstacles, high elevations, going through cover, and low elevations.

Execution time: The time (in seconds) taken by the robot to complete a navigation task. Shorter execution times indicate a more efficient and responsive navigation system, making it suitable for time-sensitive applications.

Trajectory length: The distance (in meters) traveled by the robot to reach its destination. Shorter trajectories indicate more efficient navigation, while longer trajectories may suggest sub-optimal paths or difficulty navigating in a complex environment.

D. Testing Scenarios

We evaluate our *CoverNav* navigation in different environment scenarios, including normal elevation, low-high elevation, and forest and jungle environments.

- **Normal Elevation:** This refers to a terrain with a relatively flat or gently sloping surface.
- **Low Elevation:** The terrain has a normal height and features such as buildings, fences, and other minor impediments. The maximum elevation ($\leq 1\text{m}$).
- **Low-High Elevation:** The terrain includes both high and low elevations. It has elevations ranging from 1m to 3m.
- **Forest and Jungle:** A realistic forest environment that includes diverse terrain and extensive areas of cover [18]. The maximum elevation $\geq 4\text{m}$.

E. Results

In this section, we analyze the navigation performance of *CoverNav* in various testing scenarios mentioned in the previous section.

1) Navigation Performance

Fig. 4 illustrates the agent's consistent performance improvement across scenarios as episodes progress, demonstrating effective learning. In the normal elevation scenario, the robot navigated successfully, efficiently finding shelter spots. In the low elevation scenario, it handled obstacles and utilized cover effectively. The low-high elevation scenario posed increased complexity, and the robot adapted to uneven terrains with a steeper learning curve. In the forest and jungle environment, the robot faced the most challenges but gradually improved its navigation skills. Our results showcase the effectiveness of *CoverNav* in diverse and challenging terrains, highlighting the agent's learning ability in adapting to different environmental conditions.

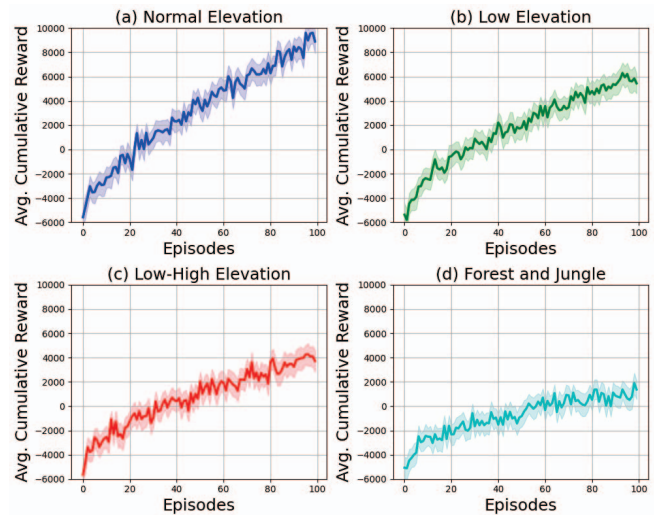


Fig. 4: Performance comparison of *CoverNav* under four scenarios: (a) Normal Elevation, (b) Low Elevation, (c) Low-High Elevation, (d) Forest and Jungle

2) Comparison with Baselines

Table I provides a comparison of the performance metrics of the DWA, GA-Nav, TERP, and *CoverNav* methods for robot navigation in various terrains. *CoverNav* demonstrates competitive performance in terms of goal-reaching and obstacle avoidance,

with success rates similar to GA-Nav and TERP. However, it is worth noting that the execution time for *CoverNav* is longer compared to the other methods, attributed to the more complex algorithms used for cover detection and following. Despite the slightly longer execution time, the superior performance of *CoverNav* in success rate and trajectory length makes it a promising method for robot navigation in various terrains, particularly in scenarios where cover following is of utmost importance.

Metrics	Method	Normal Elevation	Low Elevation	Low-High Elevation	Forest and Jungle
Success Rate (%)	DWA	72	82	67	X
	GA-Nav	79	83	71	X
	TERP	82	83	69	X
	CoverNav (Ours)	81	82	70	86
Trajectory Length (m)	DWA	5	4	6	X
	GA-Nav	6	5	7	X
	TERP	9	5	7	X
	CoverNav (Ours)	9	8	7	12
Execution Time (s)	DWA	12	14	17	X
	GA-Nav	14	13	16	X
	TERP	10	12	20	X
	CoverNav (Ours)	15	17	22	25

TABLE I: Performance Metrics Comparison of DWA, GA-Nav, TERP, and CoverNav (Ours) Methods for Robot Navigation in Various Terrains.

V. DISCUSSION

Our proposed navigation strategy is effective in outdoor environments, showing increased success with more episodes played, across different terrains and elevations. Our analysis highlights the importance of training episode length, complexity, and goal distance. Specifically, longer-distanced goals (e.g., 50 meters) caused robot instability, leading to collisions and non-convergence. Strategies to mitigate this include adjusting the reward function, fine-tuning the training process, and gradually increasing goal difficulty. This progressive approach builds robust navigation abilities and balances training difficulty with stability. The current implementation is offline, necessitating further real-time validation in unpredictable environments. Although Unity-based evaluations offer insights, real-world validation is essential for confirming effectiveness. A small-scale, real-world test would affirm its applicability and bridge the gap between simulated and actual scenarios.

VI. CONCLUSION AND FUTURE WORK

We presented a novel DRL-based method for guiding a robot in unstructured terrain environments, prioritizing maximum cover for enhanced safety and covert capabilities. Despite potentially longer trajectories compared to existing approaches, our method offers the advantage of actively seeking and utilizing cover, crucial in contested environments. Future work includes improving obstacle and cover detection by considering additional factors like geometry and dynamic reward systems and experimenting in simulated multi-robot systems [19]. Additionally, leveraging large-scale RGB datasets for training will potentially improve the model's ability to detect small objects in complex terrains.

ACKNOWLEDGMENT

This work has been partially supported by U.S. Army Grant #W911NF2120076, ONR Grant #N00014-23-1-2119, NSF CAREER Award #1750936, NSF REU Site Grant #2050999 and NSF CNS EAGER Grant #2233879. The

authors would also like to thank Avijoy Chakma, Zahid Hasan, Anuradha Ravi for their valuable feedback, Arthur C. Schang for setting up Semantic Segmentation in ARL Unity Environment, and Wanying Zhu for conducting initial experiments.

REFERENCES

- [1] Kasun Weerakoon, Adarsh Jagan Sathyamoorthy, Utsav Patel, and Dinesh Manocha. Terp: Reliable planning in uneven outdoor environments using deep reinforcement learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 9447–9453. IEEE, 2022.
- [2] Tianrui Guan, Divya Kothandaraman, Rohan Chandra, Adarsh Jagan Sathyamoorthy, Kasun Weerakoon, and Dinesh Manocha. Ga-nav: Efficient terrain segmentation for robot navigation in unstructured outdoor environments. *IEEE Robotics and Automation Letters*, 7(3):8138–8145, 2022.
- [3] Mohamed Al Marzouqi and Ray A Jarvis. Robotic covert path planning: A survey. In *2011 IEEE 5th international conference on robotics, automation and mechatronics (RAM)*, pages 77–82. IEEE, 2011.
- [4] Kai Ninomiya, Mubbasir Kapadia, Alexander Shoulson, Francisco Garcia, and Norman Badler. Planning approaches to constraint-aware navigation in dynamic environments. *Computer Animation and Virtual Worlds*, 26(2):119–139, 2015.
- [5] Kaicheng Zhang, Farzad Niroui, Maurizio Ficocelli, and Goldie Nejat. Robot navigation of environments with unknown rough terrain using deep reinforcement learning. *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–7, 2018.
- [6] Bo Zhou, Jianjun Yi, and Xinke Zhang. Learning to navigate on the rough terrain: A multi-modal deep reinforcement learning approach. In *2022 IEEE 4th International Conference on Power, Intelligent Computing and Systems (ICPICS)*, pages 189–194. IEEE, 2022.
- [7] Jinyoung Choi, Kyungsik Park, Minsu Kim, and Sangok Seok. Deep reinforcement learning of navigation in a complex and crowded environment with a limited field of view. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5993–6000. IEEE, 2019.
- [8] Utsav Patel, Nithish Kumar, Adarsh Jagan Sathyamoorthy, and Dinesh Manocha. Dynamically feasible deep reinforcement learning policy for robot navigation in dense mobile crowds. *arXiv preprint arXiv:2010.14838*, 2020.
- [9] Jumman Hossain, Abu-Zaher Faridee, Nirmalya Roy, Anjan Basak, and Derrik E. Asher. Covernav: Cover following navigation planning in unstructured outdoor environment with deep reinforcement learning. *arXiv preprint*, 2023.
- [10] Biao Gao, Shaochi Hu, Xijun Zhao, and Huijing Zhao. Fine-grained off-road semantic segmentation and mapping via contrastive learning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5950–5957. IEEE, 2021.
- [11] Amirreza Shaban, Xiangyun Meng, JoonHo Lee, Byron Boots, and Dieter Fox. Semantic terrain classification for off-road autonomous driving. In *Conference on Robot Learning*, pages 619–629. PMLR, 2022.
- [12] Viktor Wiberg, Erik Wallin, Tomas Nordfjell, and Martin Servin. Control of rough terrain vehicles using deep reinforcement learning. *IEEE Robotics and Automation Letters*, 7(1):390–397, 2021.
- [13] Shirel Josef and Amir Degani. Deep reinforcement learning for safe local planning of a ground vehicle in unknown rough terrain. *IEEE Robotics and Automation Letters*, 5(4):6748–6755, 2020.
- [14] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- [15] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, 1997.
- [16] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous robots*, 34:189–206, 2013.
- [17] Péter Fankhauser and Marco Hutter. A universal grid map library: Implementation and use case for rough terrain navigation. *Robot Operating System (ROS) The Complete Reference (Volume 1)*, pages 99–120, 2016.
- [18] Avi Spector, Wanying Zhu, Jumman Hossain, and Nirmalya Roy. Simulated forest environment and robot control framework for integration with cover detection algorithms. In *Proceedings of 9th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCA2022)*, 2022.
- [19] Emon Dey, Jumman Hossain, Nirmalya Roy, and Carl Busart. Synchrosim: An integrated co-simulation middleware for heterogeneous multi-robot system. In *18th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 334–341. IEEE, 2022.