# Tension-Aware Path Planning for Tethered Robots on Extreme Terrain Using Reinforcement Learning

**Team Member(s) :** Rahul Kumar

## 1. Introduction:

In the domain of planetary exploration, the navigation of extreme terrains poses a significant challenge for robotic systems. Tethered robots, such as Axel, can navigate extreme terrains like steep slopes on planetary missions by leveraging the tensions provided by their tethers [1], [2]. These robots often operate in pairs, with one navigating a flat surface while the other traverses steep slopes, supported by tether tension. However, ensuring optimal tension on the tether, crucial for successful exploration on steep slopes, can be tricky, especially if traction on the top robot's wheels is insufficient to support the tension required. In such cases, the tether can be wound around objects in the environment (e.g. rocks) to induce the capstan effect, where the resulting friction can augment the tension provided by the tether, enabling the tethered robot to navigate the steep slopes (Fig. 1b).

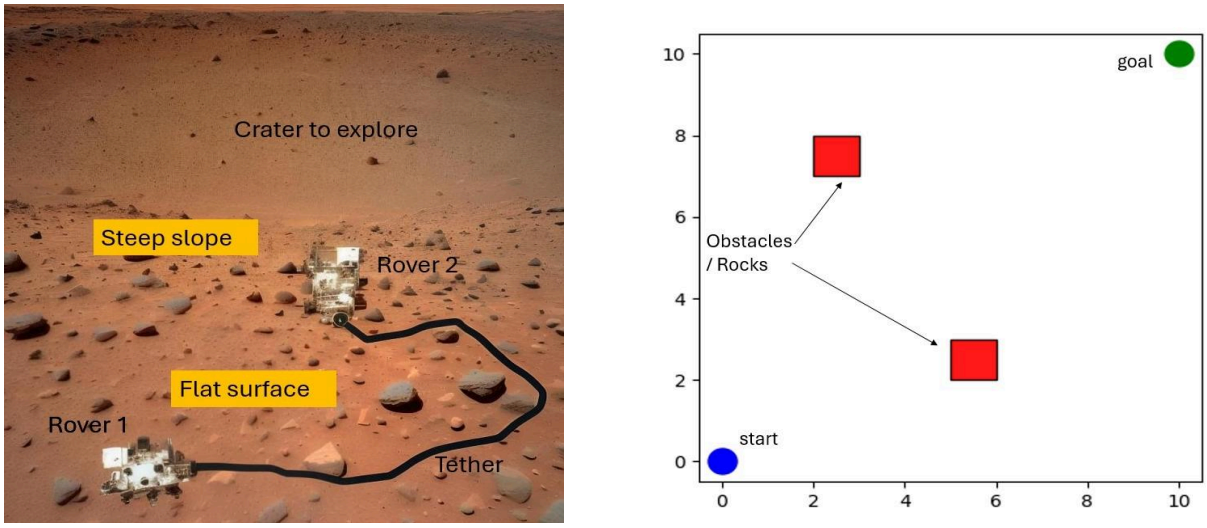
To address this challenge, this project proposes a reinforcement learning-based approach for tension-aware, winding-constrained path planning for tethered robots. The goal is to find a collision-free path from the start state to the goal state such that the user-defined winding angle is achieved to induce the required tether tension due to the capstan effect. The approach uses a variety of reinforcement learning techniques such as Monte Carlo control, SARSA, Q-learning, and Deep Q-learning [4] to learn an optimal policy for path planning. This policy considers winding constraints and optimizes the overall path, enhancing adaptability to complex environments. Through training the agent to navigate while managing tether tension, the project aims to develop a robust solution for tethered robots operating in extreme terrain conditions.

(a) (b)

Fig. 1. (a) Rover on slope supported by wire tension [1] (b) Winding of wire around rocks to augment tension using capstan effect

## 2. Problem Formulation:

The problem is formulated as a robot traversing a grid map consisting of polygonal obstacles, with the objective of moving from an initial state i $\in \mathbb{R}^2$ to a goal state g$\in \mathbb{R}^2$ while satisfying the desired winding constraint angle denoted by $\theta_d$. The robot is tethered i.e., the wire is connected to the robot, and the other end of the wire is fixed at the start state. The task is to find a path *P* from i to g, *P* (i,g) s.t. $\theta_a >= \theta_d$ where $\theta_a$ is the actual winding around the obstacles achieved when the robot reaches the goal.




(a) (b)

Fig. 2. Problem representation (a) Actual map with one rover on a flat surface and another rover going down the steep slope after winding the tether around rocks (b) Grid world map for training with an agent going from start to goal after winding around obstacles.

The path planning for the tethered robot is formulated as a Goal-augmented Markov Decision Process (GA-MDP), which is represented as a tuple (S,A,T,R,$\gamma$,G). This approach provides an efficient solution as it allows for the achievement of multiple sequential goals. In this framework, S and A signify the state space and action space of the MDP respectively. Two types of state spaces (s $\in$ S) were experimented with: the first one, S=$\mathbb{R}^2$, constitutes a tuple of (x,y), and the second one, S=$\mathbb{R}^2$x$\mathbb{R}$, constitutes a tuple (x,y,$\theta$). Here, (x,y) indicates the X and Y position of the cell in the grid map, while $\theta$ denotes the actual winding angle.

The agent can only visit the cells where there are no obstacles present. As for the action space (a $\in$ A), A = {Left, Right, Up, Down}, which means that the agent can choose one of these actions at the current state s to move to a new state s'. The transition function, T(s,a,s') = P(s'|s,a), provides the probability of ending up in s' after taking an action a in state s. However, given the consideration of a model-free environment, this transition function isn't directly available to the agent. Instead, expectations are approximated by averaging over samples, which represent the state-value or action-value functions.

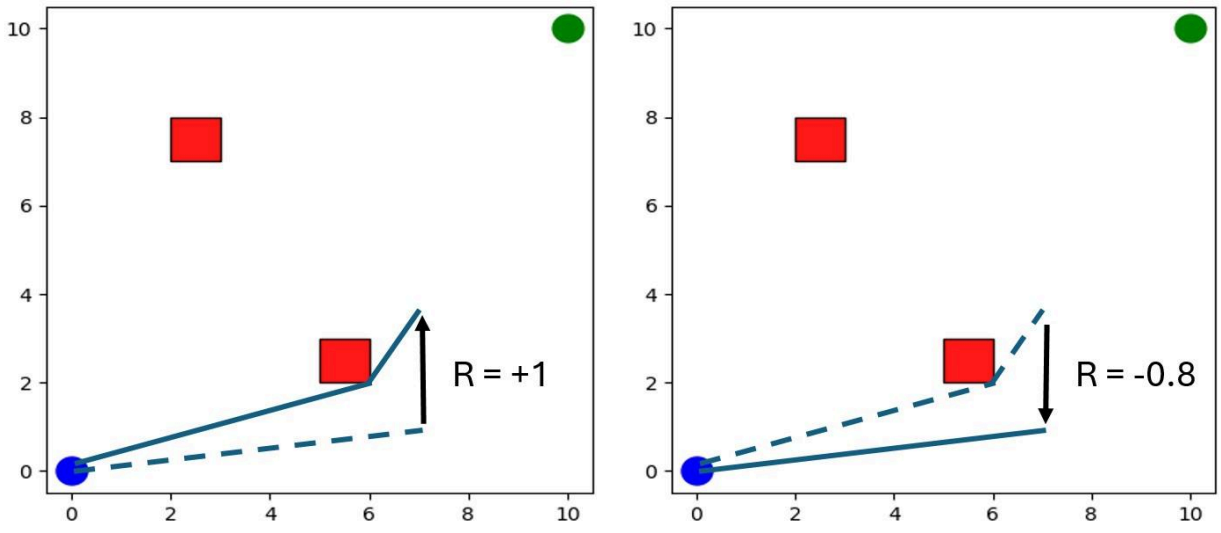
The framework also includes $\gamma$, which is a discount factor for the return, and G, which denotes the goal space, describing the tasks that are winding around obstacles and then reaching the goal. This problem framework, thus, aims at achieving the sequential goals in tension-aware, winding-constrained path planning for tethered robots in a robust and efficient manner.

R : S x A x G→ $\mathbb{R}$ is the reward function, which generates a scalar value when action a is taken depending on the current state s and current goal status. A policy π : S x G→A defines the agent's action selection criterion for taking action a ∈ A for each state s ∈ S and G. Various RL algorithms have been adopted to optimize the policy π to achieve the best expected return from any initial state. Reward shaping [3] is used to optimize the path while satisfying the winding constraint. First, the goal of satisfying the winding angle is checked, if the current winding angle is less than the desired winding angle, then the current winding angle is checked with respect to the previous step winding angle. If found larger, the reward is +1 but if found smaller then a reward of -0.8 is given. In case, the winding angle does not change, then the reward will be -0.1. These reward allocations ensure the learning of the agent at each step by allocating dense rewards. Once the first goal is achieved, a negative reward is provided corresponding to the current Euclidean distance from the goal state. If both goals are achieved before truncation (in this case maximum step in an episode is 500), reward of 10 is allocated.

Algorithm for reward function:

```
reward(current winding, previous winding, desired winding, agent pos, goal pos)
    If current winding < desired winding, then
        If current winding > previous winding, then
            Reward → 1
        Else if current winding < previous winding, then
            Reward → -0.8
        Else
            Reward → -0.1
    Else
        Reward → - Euclidean distance to goal(agent pos, goal pos)

    If current winding >= desired winding and agent pos = goal, then
        Reward → 10
```




(a) (b)

Fig. 3. Reward allocation for winding and unwinding (a) Winding around the obstacle at each step result in R+1 (b) Unwinding at each step result in R-0.8

## 3. Experiments and Results:

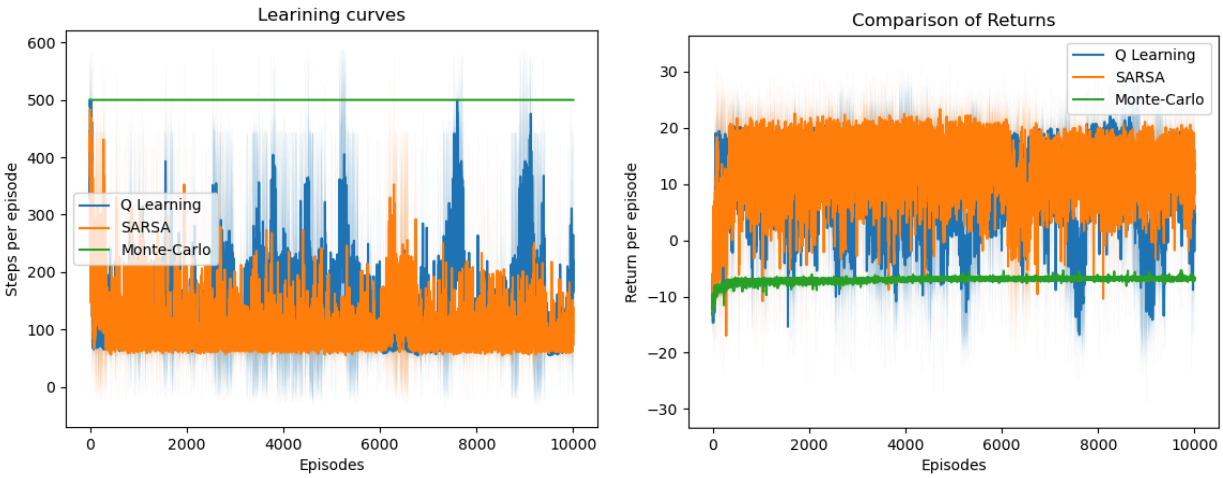
To tackle the problem of tension-aware, winding-constrained path planning for tethered robots, several reinforcement learning algorithms were implemented and evaluated. These include on-policy Monte Carlo (MC) control, SARSA, Q-learning, and Deep Q-learning. Experiments were conducted with MC, SARSA, and Q-learning on both state spaces and Deep Q-learning on the second state space.

### 3.1. MC, SARSA, and Q-learning with State Space (x,y)

In this trial, the x and y position of the agent is considered as a state, and 3 RL algorithms are used to train the agent. The training is done for 10000 episodes with a maximum episode length truncated at 500 steps. The discount factor is taken as 0.99 and the step size is 0.3.

For the initial set of trials, the epsilon value was kept constant at 0.2, and the agent was trained using all three algorithms. As shown in Fig. 4, the results indicated that Q-learning and SARSA performed well, while Monte Carlo did not yield satisfactory results.

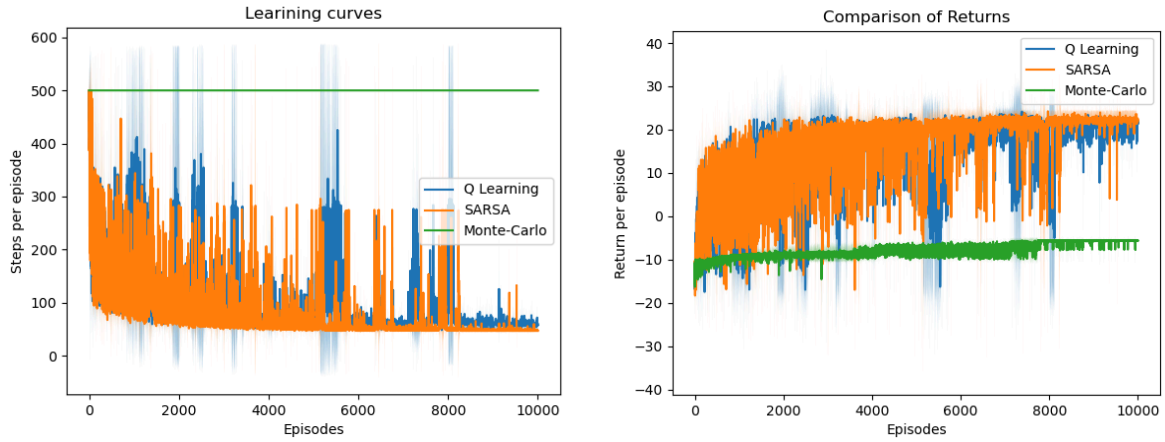
After analyzing the plots, it is observed that more initial exploration can help in training further, and hence exponential scheduler is used for varying the epsilon value from 0.5 to 0.01. Fig. 5 shows the results with variable exploration. The performance for Q-learning and SARSA is improved and variance is also decreased. Monte Carlo still failed to perform.




(a) (b)

Fig. 4. Agent is trained with On-Policy Monte-Carlo, SARSA, and Q-learning with gamma 0.9, step size 0.3, and epsilon 0.2 (a) Learning curve comparison plot (b) Return comparison plot

The path is visualized for agents trained with Q-learning as it has the best performance. It is clear from Fig. 6 that the agent can learn to wind around obstacles and reach the goal as well. But with state consisting of XY position doesn't capture the essence of goal-conditioned RL as the agent is going to goal position and then returning to complete the winding and then again going to goal position. This behavior of the agent is not desired, the agent first should learn to wind around obstacles and then only reach the goal state.




(a) (b)

Fig. 5. Agent is trained with On-Policy Monte-Carlo, SARSA, and Q-learning with gamma 0.9, step size 0.3, and exponentially varying epsilon 0.5->0.01 (a) Learning curve comparison plot (b) Return comparison plot

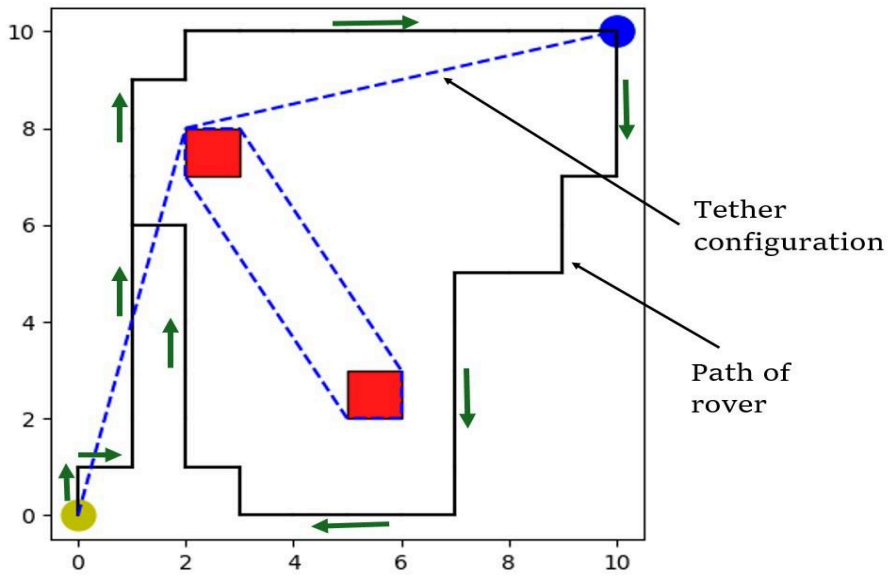



Fig. 6. Path visualization of rover trained on Q-learning with state space (x,y), 58 steps taken in this trial to achieve 2*Pi winding angle.

### 3.2. MC, SARSA, and Q-learning with State Space (x,y,θ)

As described above, only the x and y positions are not enough to capture the complete information, so the current winding angle is added to the State Space. With this new formulation, the agents are trained again on all 3 algorithms with variable exploration. Now agent learns to wind first and then only move to the goal position. Fig. 6 provides the path visualization with different winding angle requirements.

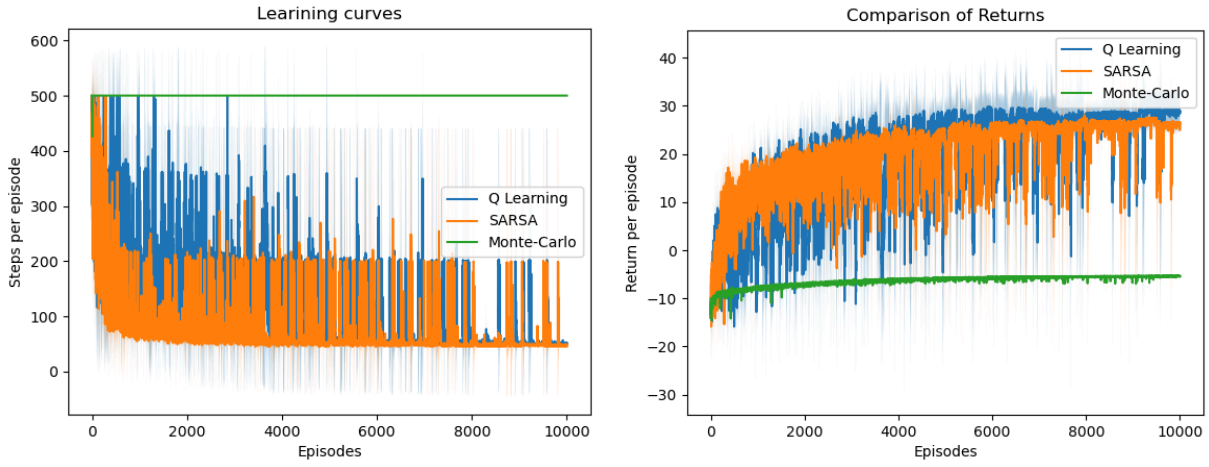
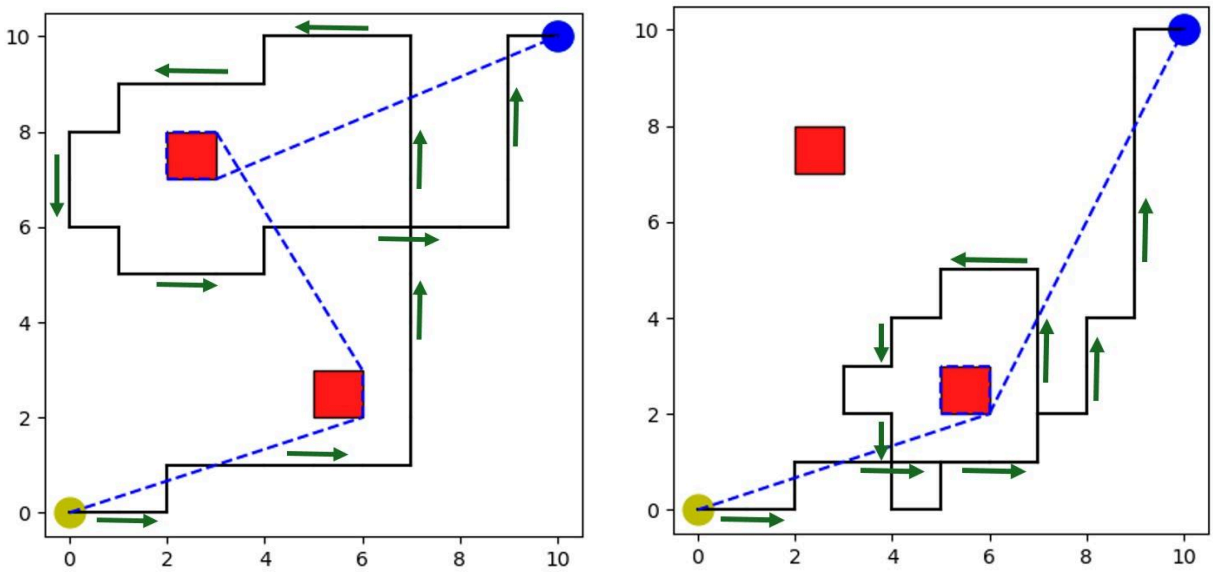



Fig. 7. Agent is trained with State Space (x,y,θ) with gamma 0.9, step size 0.3, and exponentially varying epsilon 0.5->0.01 (a) Learning curve comparison plot (b) Return comparison plot



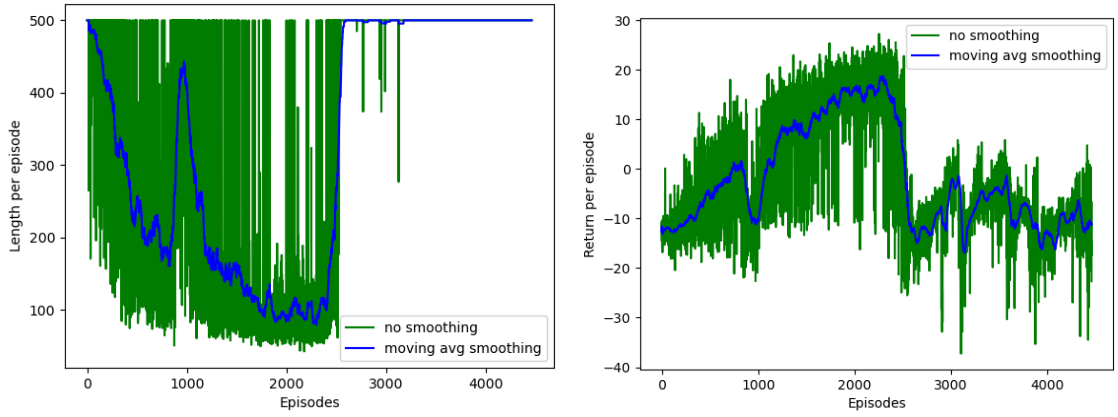
(a) (b)

Fig. 8. Path visualization of rover trained on Q-learning with state space (x,y,θ) (a) Desired winding angle is 2*Pi, 44 steps taken (b) Desired winding angle 1.5*Pi, 38 steps taken

### 3.3. Deep Q-learning with State Space (x,y,θ)

Deep Q-learning is also employed with a state space defined as (x, y, θ). To manage exploration, an exponential scheduler was employed, gradually reducing exploration from 1 to 0.01 over time. Experience replay memory of 200k was utilized, and the agent was trained for 1.5 million steps.

However, the obtained results fell short of expectations. Initially, during the early episodes when exploration was high, the agent displayed promising learning behavior. However, as exploration decreased, the agent's learning stagnated, failing to converge. Fig. 9 shows the learning behavior of the agent for Deep Q-learning.




(a) (b)

Fig. 9. Deep Q-learning performance (a) Learning curve (b) Return plot

## 4. Conclusion

In this project, a Goal-Conditioned Markov Decision Process (GA-MDP) was formulated for the path planning of tethered robots in a grid world consisting of polygonal obstacles. The performance of three reinforcement learning (RL) algorithms, namely Monte-Carlo control, SARSA, and Q-learning, was compared to learn the task of winding around obstacles and reaching the goal. The performance of the algorithms was evaluated through simulation. The results showed that Q-learning performed the best and was able to consistently find a path that satisfies the user-defined winding constraint if such a path exists. Furthermore, a Deep Q-learning approach was implemented in its standard form. However, it did not perform as well as the traditional Q-learning algorithm, suggesting that further tweaking and optimization of the Deep Q-learning algorithm may be required to achieve good performance for this specific problem. Overall, a complete solution for winding-constrained path planning tasks was proposed, utilizing the Q-learning algorithm as the most effective approach among the methods evaluated. The GA-MDP formulation and the reward shaping technique employed proved to be effective in guiding the agent towards the desired behavior of winding around obstacles while reaching the goal. The findings of this project provide valuable insights into the application of RL techniques for solving complex path-planning problems with additional constraints, such as the winding requirement.

**References:**


[1] P. Abad-Manterola, I. A. D. Nesnas and J. W. Burdick, "Motion planning on steep terrain for the tethered axel rover," *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011, pp. 4188-4195, doi: 10.1109/ICRA.2011.5980238.

[2] M. Paton, M. P. Strub, T. Brown, R. J. Greene, J. Lizewski, V. Patel, J. D. Gammell, and I. A. Nesnas, “Navigation on the line: Traversability analysis and path planning for extreme-terrain rappelling rovers,” in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020, pp. 7034–7041.

[3] Liu, ZiXuan & Wang, Qingchuan & Yang, Bingsong. (2022). Reinforcement Learning-Based Path Planning Algorithm for Mobile Robots. Wireless Communications and Mobile Computing. 2022. 1-10. 10.1155/2022/1859020.

[4] Aleksandr I. Panov, Konstantin S. Yakovlev, Roman Suvorov, Grid Path Planning with Deep Reinforcement Learning: Preliminary Results, Procedia Computer Science, Volume 123, 2018, Pages 347-353, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2018.01.054.

[5] Liu, M., Zhu, M., and Zhang, W., “Goal-Conditioned Reinforcement Learning: Problems and Solutions”, arXiv e-prints, 2022. doi:10.48550/arXiv.2201.08299.

[6] Gao, Junli & Ye, Weijie & Guo, Jing & Li, Zhongjuan. (2020). Deep Reinforcement Learning for Indoor Mobile Robot Path Planning. Sensors. 20. 5493. 10.3390/s20195493.