# 16-831 RL Project Proposal

**Daniel Yang**
danielya
danielya@andrew.cmu.edu

**Yu-Hsuan Li**
yuhsuanl
yuhsuanl@andrew.cmu.edu

**Ricky Yuan**
rickyy
rickyy@andrew.cmu.edu

**Rena Ju**
renaju
renaju@andrew.cmu.edu

**Ethan Lai**
ethanlai
ethanlai@andrew.cmu.edu

**Abstract:** Bouldering is a form of rock climbing that involves intelligent trajectory planning and intricate motor control. In contrast to classic humanoid learning tasks such as Humanoid Standup, bouldering presents unique physical constraints specific to each climbing route, significantly restricting the solution space.

In this course project, we aim to develop a virtual agent that learns to boulder from scratch, adapting dynamically to each unique boulder problem. We plan to begin with simple shapes as climbing holds and progress to the general case of 3D-scanned meshes, which exist in a higher dimensional space. Furthermore, the agent will be designed to adapt to unseen boulder problems in a few-shot manner in the future. This proposal outlines the motivation behind our work, the potential applications of such a system, and the scenario settings and methods we propose to achieve these goals.

**Keywords:** Robot Learning, Few-shot adaptation, Humanoid, Motion Planning

## 1 Introduction

Rock climbing is becoming an increasingly popular sport. It demands a combination of balance, coordination, finger and upper body strength, footwork, technical knowledge, and strategic planning. This sparked our interest in exploring how a robot learns to climb without expert demonstration. We can break down the problem into several key components. Given a boulder problem, a climber first inspects the route and proposes a climbing sequence, known as the *'beta'* in climbing jargon. After selecting a candidate sequence, the climber must execute these movements with precision. However, due to physical differences such as height, arm span, and finger strength, not all climbers can perform the same movements. This disparity requires the climber to adapt and develop a personal style that best suits itself and generate a *'beta'* accordingly.

Training a virtual agent to climb has significant implications for both boulder training and disaster relief. For bouldering, it could serve as a dynamic tutorial for beginners or provide tailored guidance for elite climbers, offering personalized strategies based on different body types. The agent could also help athletes break out of learned patterns, offering fresh approaches to difficult routes. Moreover, this research could contribute to the development of objective metrics for evaluating the difficulty of bouldering problems, which currently lacks a standardized measure. Beyond sport, such an agent could assist in disaster scenarios where human access is limited, navigating complex environments to perform tasks that would be dangerous for humans.

In real-world applications, this technology could be valuable in disaster relief efforts. Imagine a robot capable of climbing reaching people trapped by landslides or in remote areas faster than humans could, potentially providing crucial assistance in rescue operations.

## 2 Method

### 2.1 Scenario

We define our training scenarios as follows:

1. **Environment**: A simplified humanoid robot and a climbing wall, where the robot has perfect information about the climbing route.
2. **Climbing Wall**: Initially, we design the climbing holds to be hexagonal, following the approach outlined by Nguyen and Shimada [1], which facilitates the robot's ability to easily grasp and step on them. However, we aim to progress to a more general case, where both the wall and climbing holds are represented as 3D meshes.
3. **Robot Configuration**: The robot's hands are equipped with hooks for gripping the holds, while its feet are flat for stability.
4. **Initial State**: The robot starts on a designated climbing hold, eliminating the need to navigate its way to the first hold.
5. **Goal**: To reach a predetermined climbing hold on the wall, with minimum physical exertion.

### 2.2 Proposed Method

Some previous work has dedicated efforts to solving rock climbing problems. For instance, Focchi et al. [2] proposed a method for climbing real mountains; however, it requires drilling holes, which is not suitable for bouldering since we aim to avoid damaging the walls. Similarly, Naderi et al. [3] proposed methods for bouldering, but their work assumes the use of ball-and-socket joints, which may not be realistic for actual robots. Additionally, Nguyen and Shimada [1] developed a realistic robot to address climbing problems using forward and inverse kinematics; however, their approach was limited to simple ladder climbing scenarios and may not be applicable to more complex situations.

We have identified two potential methods to achieve the climbing task for the robot:

#### 2.2.1 Method 1: End-to-End Training

We plan to train the agent in an end-to-end manner utilizing DDPG proposed by Lillicrap [4] or PPO proposed by Schulman et al. [5]. The reward function can be defined as follows:

- A fixed reward for successfully reaching the top
- Partial rewards for reaching specific waypoints
- Negative rewards based on the cumulative muscle exertion over time.

And the states follow the default state space from the Gymnasium *Humanoid* environment setup.

In contrast to standard PPO, our approach includes an additional input: the climbing route. Each climbing route is unique to its respective problem but remains static throughout each epoch. This setup resembles a conditional generative model, where the route acts as a condition that influences the distribution of action probabilities. As a result, different climbing routes lead to different action choices, even with the same policy and state.

The standard policy $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$ calculates action probability distribution solely based on $\mathbf{s}_t$; whereas our policy takes in an additional conditional input $\mathbf{c_i}$, calculated from the encoding [6] of the $i$'th boulder route mesh $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t, \mathbf{c_i})$

#### 2.2.2 Method 2: Two-Stage Process

The second approach divides the entire process into two stages in a coarse-to-fine manner:

**Stage 1: Path Computation** In the first stage, the robot computes the optimal path to the goal using its knowledge of the rock positions. This algorithm needs to take into account the arm and leg lengths of the robot. We are considering using a diffusion model for path generation, similar to the one provided by Carvalho et al. [7].

**Stage 2: Path Following** In the second stage, the robot will move to follow the desired path. At this point, the robot only needs to consider the target holds, allowing it to concentrate solely on its immediate goals without being distracted by distant holds. The implementation could resemble that described in 2.2.1, but this time the conditions will pertain only to a portion of the boulder route mesh.

## 3 Plan

1. Create a simplified virtual agent model in MuJoCo.
2. Design representations of climbing holds and generate climbing wall environments.
3. Define rewards and observation space (including normal vectors of climbing holds, forces applied on each hold, etc.).
4. Implement our robot learning algorithm.
5. Implement adaptation mechanisms for few-shot learning.

## References


[1] D. A. Nguyen and A. Shimada. Humanoid climbing robot modeling in matlab-simechanics. *IECON2013*, pages 4160–4165, 2013.

[2] M. Focchi, A. Del Prete, D. Fontanelli, M. Frego, A. Peer, and L. Palopoli. Alpine: a climbing robot for operations in mountain environments. *arXiv preprint arXiv:2403.15142*, 2024.

[3] K. Naderi, J. Rajamäki, and P. Hämäläinen. Discovering and synthesizing humanoid climbing movements. *ACM Transactions on Graphics (TOG)*, 36(4):1–11, 2017.

[4] T. Lillicrap. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[5] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[6] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (ToG)*, 38(4):1–12, 2019.

[7] J. Carvalho, A. T. Le, M. Baierl, D. Koert, and J. Peters. Motion planning diffusion: Learning and planning of robot motions with diffusion models. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1916–1923. IEEE, 2023.