Universiteit
Leiden
The Netherlands

Bachelor Computer Science & Datascience and Artificial Intelligence

Reinforcement Learning for Minimum-Energy Path Discovery on Low-Dimensional Potential Energy Surfaces

Aleksandra Bobrova

Thomas Moerland & Felix Kleuker

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

01/06/2026

Abstract

Many molecular processes, such as chemical reactions and protein folding, proceed along a minimum energy path (MEP) on a potential energy surface (PES), and finding this path is a central problem in computational chemistry. Established methods such as the nudged elastic band (NEB) rely on deterministic optimization procedures and require the endpoints in advance, while the few existing reinforcement learning (RL) approaches either fail to reach the target minimum or recover only the transition state. The main obstacle is the design of the reward function, since its optimal solution must match the path that a chemical reaction is most likely to follow in physical reality. This bachelor thesis studies how the reward should be formulated for an RL agent to recover the MEP on the two-dimensional Müller-Brown potential. We compare three reward designs, two from earlier work that reward the negative energy of the next state and the negative energy difference between consecutive states, and a custom design that penalizes only uphill moves. The studied methods are Value Iteration that gives the exact solution on a discretized grid and PPO that learns a policy in the continuous setting. With VI, each reward is tested across the discount factors 0.99, 0.999, and 1. The results are compared against an NEB reference. The two existing rewards rely on discounting to reach the target and collapse to a shortcut that misses the saddle points, whereas the custom reward recovers the transition states and matches the NEB activation barrier, but only when the discount factor is at or near 1. Trained with this reward at $\gamma = 1$, four of five PPO seeds recover the activation energy within rounding of the reference. The custom reward is a clear improvement over the existing designs, although it leaves the descent underdetermined and the broader problem open. The code and the custom environment is available at https://github.com/Sasha580/MEPRL_thesis.git.

Contents

1 Introduction

One of the main goals in studying molecular processes is to understand the transitions between different configurations of a system. These transformations include chemical reactions, drug binding and protein folding, which are studied using computer simulations in practice. In such simulations, each configuration of the system is assigned an energy value, which forms the potential energy surface (PES), where stable configurations correspond to low-energy regions separated by barriers. This creates many possible routes that the system can take during a transformation, among which the minimum energy path (MEP) represents the most relevant transition pathway. Finding this path is essential to support the development of new materials, medicines, and technologies [1, 2, 3].

The current approaches for computing the MEP include Nudged Elastic Band (NEB) and the String Method, which rely on deterministic optimization procedures [4, 5, 6]. While being used in practice, these methods have several important limitations. First, both the initial and final minima must be known or approximated in advance, even though this is often the objective of the study [7]. This thesis proposes reinforcement learning (RL) methods, which require only a starting configuration. Second, these methods scale poorly with the dimensionality of the system, because they require a good guess of the initial path, which is hard to provide in high dimensions [8]. In contrast, RL learns a policy by sampling many trajectories, which lets it search the space of paths more broadly rather than refining one candidate.

RL methods are a subset of machine learning algorithms, designed to learn an optimal policy through a sequence of decisions [9]. To apply RL, the task is formulated as a sequential decision problem, where an agent is guided by local information to repeatedly select a move through the configuration space. This setting addresses both limitations above. One could run NEB from many initial paths to escape a poor guess, but this gives no guarantee of covering the relevant region of the surface, and each energy evaluation on the landscape is expensive. RL instead spends a single training budget on searching the space of paths directly. It also differs from supervised learning, which would need a dataset of optimal paths from some other method before training. RL needs no such ground truth, since it discovers paths itself through sampling and reward. The central difficulty is then to design a reward function that guides the agent according to physical reality.

In the field, there are very few papers that have applied RL in this setting. For example, the work of Pal et al. [10] attempted this on the benchmark Müller–Brown potential [11] but failed to find the target minimum, whereas the research of Mills et al. [12] successfully located the transition state on the same surface but did not attempt to recover the full path. Therefore, the question of whether RL can find the true meaningful path even on the toy two-dimensional Müller-Brown potential remains open. In this thesis, we are addressing this gap by investigating how the reward function should be formulated for an RL agent to successfully find the MEP.

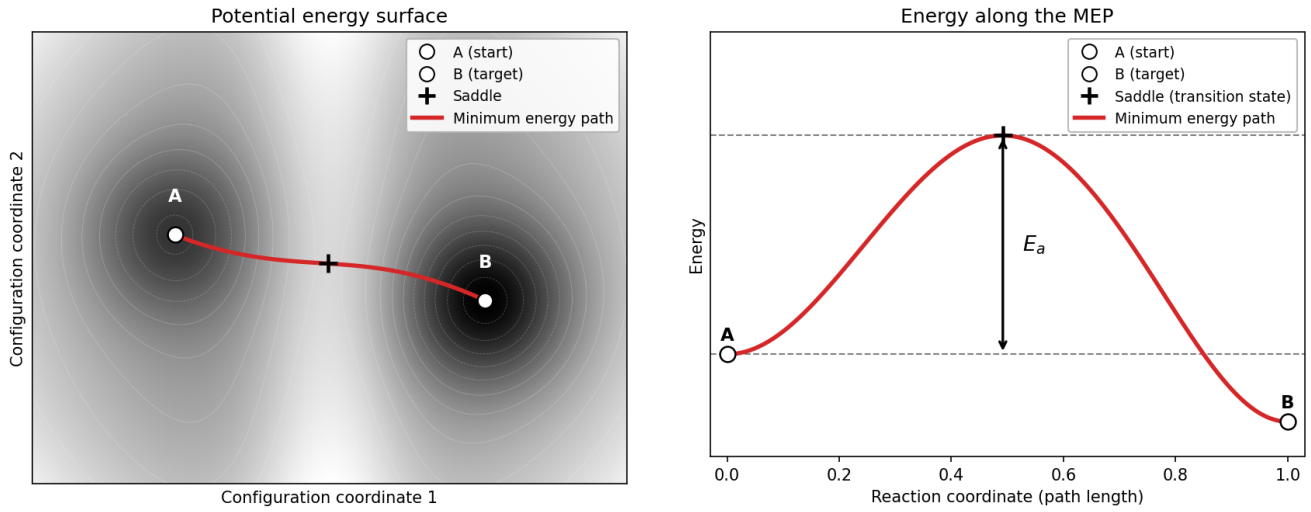
In this work, the question is studied on the Müller–Brown potential using two algorithms: Value Iteration (VI) [13] on a discretized version of the environment and Proximal Policy Optimisation (PPO) [14] in the corresponding continuous setting. We systematically compare three reward function formulations on the discretized version of the environment. The first two reward functions include the formulations used in the above-mentioned RL papers, whereas the third one is the custom reward design which does not punish an agent for going downhill. Moreover, we evaluate the effect of the discount factor γ on path quality across designs and then choose the best setting to use together with the PPO solution. The resulting paths for both algorithms are benchmarked against the reference MEP computed by the NEB method.

The results show that the custom reward function is the only formulation capable of recovering the transition states of the system, even though its resulting path avoids rolling downhill towards the intermediate basin. The final trajectory of this reward is extremely sensitive to the choice of γ and only finds the target when γ is at or near 1. In contrast, the policies under the other two rewards converge to the shortest path under any $\gamma < 1$. Therefore, they rely on discounting too much and under $\gamma = 1$ they are unstable. Although the custom reward does not solve the problem entirely, this formulation is a significant improvement compared to the existing designs in the field.

This bachelor thesis was created under the supervision of Thomas Moerland and Felix Kleuker. This chapter contains the introduction; Section 2 includes the definitions and preliminary knowledge; Section 3 discusses related work; Section 4 describes the studied methods; Section 5 includes experiments and their outcome; Section 6 interprets the results and gives more general overview of the topic; Section 7 concludes. Appendix A includes additional result, which would not fit into the main text of the report.

2 Background

This section introduces the concepts that this thesis builds on. The energy landscape is described first, which is a place where transitions happen. Then, the reinforcement learning (RL) framework is used to describe the search for that path existing on the surface.



(a) (b)

Figure 1: Schematic illustration of a PES and the MEP connecting two of its minima. **(a)** Top-down view of a schematic PES, where darker regions denote lower energy. The two minima A (start) and B (target) are separated by a single first-order saddle point, and the MEP (red) connects them over that saddle. **(b)** Energy along the MEP as a function of normalized path length. The highest point is the transition state, and the activation energy E_a is its energy difference from the starting minimum A.

2.1 Minimum Energy Paths

Potential Energy Surface. Each configuration of atoms in a molecular system can be described by a vector $\boldsymbol{s} \in \mathbb{R}^{3n}$, where n is the integer number of atoms and each contributes three spatial coordinates. This vector is mapped to a scalar energy value through a function $E : \mathbb{R}^{3n} \to \mathbb{R}$, which defines a potential energy surface (PES). In other words, every configuration of atoms has an associated energy, and physical systems tend to evolve toward configurations of lower energy. Therefore, the PES captures how likely a configuration is, and how readily the system moves from one configuration to another. Whereas the configuration space is $3n$-dimensional in reality, the concepts are easiest to picture on a two-dimensional example, as depicted in Figure 1a. Its two coordinates are abstract and do not point to specific atoms, because there is no integer n to represent such a system.

Stationary points of the PES satisfy $\nabla E(\boldsymbol{s}) = 0$, and thus, correspond to configurations where the net force on each atom vanishes. Among these, configurations with positive curvature in all directions correspond to local minima and represent stable states of the system. The examples of the two minima on the surface can also be seen in Figure 1a. Molecular systems tend to occupy these low-energy configurations, while transformations between them, such as chemical reactions and conformational changes, require the system to move through higher-energy regions of the PES.

Minimum Energy Path. Given two local minima A and B on the PES, there are in general many possible paths connecting them. Among these, the minimum energy path (MEP) is the curve ϕ^* whose tangent vector is parallel to the energy gradient ∇E at every point of the route. The red curve in Figure 1a traces such a path, but the picture is simplified to consist of only two minima. Formally, the MEP $\phi^* : [0, 1] \to \mathbb{R}^{3n}$ satisfies

$$\nabla E(\phi^*(\alpha)) \parallel \frac{d\phi^*}{d\alpha}, \quad \forall \alpha \in [0, 1], \tag{1}$$

with boundary conditions $\phi^*(0) = A$ and $\phi^*(1) = B$ [15]. This condition implies that the MEP passes through one or more first-order saddle points, each of which is called a *transition state* of the system [5]. In particular, a sequence of such transition states is involved only when there exist intermediate minima between A and B. Then, the energy difference between the starting minimum A and the highest saddle point along the way is called the *activation energy* or *activation barrier*,

$$E_a = \max_{\alpha \in [0,1]} E(\phi^*(\alpha)) - E(A), \tag{2}$$

which reflects how easily the transformation occurs. This quantity is shown in Figure 1b, where the energy is plotted along the MEP and E_a is the gap between the transition state and A.

In general, multiple paths satisfying the gradient condition may connect two minima on the PES. In practice, however, the MEP of interest is the pathway with the lowest activation energy, since the probability p of observing a transition scales as $p \propto e^{-\Delta E / k_B T}$ and the lowest-barrier route is therefore the one most likely to be observed. In this formula, k_B is the Boltzmann constant and T is the absolute temperature of the system.

Nudged Elastic Band. The *nudged elastic band* (NEB) method is the established approach for finding the MEP between two minima on a PES [4]. It represents the path as a chain of intermediate

configurations, called *images*, between the fixed start and target states. The adjacent images are connected by artificial springs that form an elastic band, explaining the name given to the method.

The algorithm proceeds iteratively, moving every image toward the MEP by a deterministic optimization step. In particular, at each update, for every image, the force acting on it is calculated. However, that force comes from a projection that keeps only two of the available components. The first is the component of the true PES force perpendicular to the path, which relaxes the image onto the MEP. The second is the component of the spring force parallel to the path, which keeps the images evenly distributed along it. This projection is the *nudging*, and it prevents the springs from cutting corners and the PES force from sliding the images down into the minima.

This thesis applies the climbing-image variant of NEB [5], where the highest-energy image is guaranteed to lie exactly on the saddle point. This allows the activation energy E_a to be read off the band precisely as the energy difference between the saddle and the initial minimum rather than estimating it. In this work, NEB is used to provide the reference MEP on low-dimensional landscapes, against which the trajectories produced by the other algorithms are compared.

2.2 Reinforcement learning

Markov Decision Process. A Markov Decision Process (MDP) is a framework for modeling sequential decision making. It is defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is the set of possible states, $\mathcal{A}$ is the set of available actions, $\mathcal{T}(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a})$ is the transition function giving the probability of reaching state $\boldsymbol{s}'$ after taking action $\boldsymbol{a}$ in state $\boldsymbol{s}$, and $\mathcal{R}(\boldsymbol{s}, \boldsymbol{s}')$ is the reward signal which the agent receives for that transition. The discount factor $\gamma \in [0, 1]$ weighs the value of future rewards, where values closer to 1 force an agent to prefer long-term returns over the current ones.

One of the main assumptions of MDP is that future gains depend only on the current state and action, not on the sequence of events that preceded them. In particular, an agent interacts with the MDP by observing the current state, selecting an action, and receiving a reward. This process is repeated in each state until the termination conditions are met. The goal of the agent is to learn a policy π:

$$\pi : \mathcal{S} \times \mathcal{A} \to [0, 1], \qquad \sum_{\boldsymbol{a} \in \mathcal{A}} \pi(\boldsymbol{a} \mid \boldsymbol{s}) = 1 \quad \forall \boldsymbol{s} \in \mathcal{S}. \tag{3}$$

that maximizes the discounted *cumulative reward*, namely return $G_{1:N}$, over time. The return from timestep t is defined as follows:

$$G_{t:N} = \sum_{k=t}^{N} \gamma^{k-t} \mathcal{R}(\boldsymbol{s}_k, \boldsymbol{s}_{k+1}),$$

where N is the *episode length* or *horizon* that can be infinite, and $\mathcal{R}(\boldsymbol{s}_k, \boldsymbol{s}_{k+1})$ is the reward received at time step k of the episode. The policy can depend on some parameter vector θ, which can be denoted as π_θ. To define optimality, the action-value function $q^\pi(\boldsymbol{s}, \boldsymbol{a})$ is introduced. It is defined as the expected return obtained by taking action $\boldsymbol{a}$ in state $\boldsymbol{s}$ and following π afterwards, where the expectation is over the policy and the transition dynamics $\mathcal{T}$,

$$q^\pi(\boldsymbol{s}, \boldsymbol{a}) = \mathbb{E}_{\pi, \mathcal{T}} \left[\sum_{k=t}^{\infty} \gamma^{k-t} \mathcal{R}(\boldsymbol{s}_k, \boldsymbol{s}_{k+1}) \,\middle|\, \boldsymbol{s}_t = \boldsymbol{s},\ \boldsymbol{a}_t = \boldsymbol{a} \right] = \mathbb{E}_{\pi, \mathcal{T}} [\, G_{t:\infty} \mid \boldsymbol{s}_t = \boldsymbol{s},\ \boldsymbol{a}_t = \boldsymbol{a} \,].$$

The agent seeks an optimal policy π^* that maximizes this value in every state,

$$\pi^* = \arg\max_{\pi} q^{\pi}(\boldsymbol{s}, \boldsymbol{a}), \quad \forall \boldsymbol{s} \in \mathcal{S}, \boldsymbol{a} \in \mathcal{A}.$$

In this thesis, the MDP framework is used to frame MEP search as a sequential decision problem, where the state is the current molecular configuration and actions are moves through configuration space. The main challenge in this formulation is to design a reward function that encourages the agent to find a MEP-like path from A to B. Since the MEP objective itself does not depend on time, a reward function that encodes this objective should ideally remain well-defined at $\gamma = 1$. As a result, the discount factor should act as an additional design choice rather than a convergence requirement.

Policy Gradient Methods. Policy Gradient methods are a class of RL algorithms for solving MDP problems. They optimize the policy π_θ directly rather than learning a value function to derive it. In particular, the method updates the parameters θ through gradient ascent to maximize the expected cumulative reward $J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[G_{1:N}]$, where τ is an episodic trajectory drawn from the policy. The gradient of $J(\theta)$ with respect to the parameters is

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=1}^{N} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_t \mid \boldsymbol{s}_t)\, G_{t:N}\right], \tag{4}$$

where $G_{t:N}$ is the discounted reward received from step t to the end of the episode.

Weighting each term by the return $G_{t:N}$ is unbiased but has high variance, because the return of a single trajectory fluctuates strongly from one roll-out to the next. *Actor-critic* methods reduce this variance by learning a value function alongside the policy, where the policy is the *actor* and the value function is the *critic*. The critic estimates the state value function $v^{\pi_\theta}(\boldsymbol{s})$ that is the expected return from a state when following the policy. In the following text, the estimate is represented as $V^{\pi_\theta}(\boldsymbol{s})$. The true value function is used to calculate an *advantage* term $A^{\pi_\theta}(\boldsymbol{s}_t, \boldsymbol{a}_t)$:

$$A^{\pi_\theta}(\boldsymbol{s}_t, \boldsymbol{a}_t) = q^{\pi_\theta}(\boldsymbol{s}_t, \boldsymbol{a}_t) - v^{\pi_\theta}(\boldsymbol{s}_t), \tag{5}$$

where $q^{\pi_\theta}(\boldsymbol{s}_t, \boldsymbol{a}_t)$ is the expected return from taking action $\boldsymbol{a}_t$ in $\boldsymbol{s}_t$ and following the policy afterwards. The advantage term describes how much better or worse an action is than the policy's average behavior at $\boldsymbol{s}_t$, and can be plugged into the formula of the gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=1}^{N} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_t \mid \boldsymbol{s}_t)\, A^{\pi_\theta}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]. \tag{6}$$

Because v^{π_θ} depends only on the state, subtracting it leaves the gradient unbiased while reducing the variance of its estimate, so each computed gradient is less noisy.

3 Related Work

The problem of identifying MEPs and transition states on PESs has long been studied in computational chemistry and molecular physics. The established methods approach it through deterministic

optimization procedures, whereas more recent solutions apply machine learning (ML) to the problem. This thesis belongs to the second group of research and uses reinforcement learning (RL) algorithms to find the path directly. The deterministic method, on the other hand, is accepted as a ground truth on the low-dimensional surface for the studied solutions.

Deterministic path optimization. The standard way to obtain a MEP is to represent the path as a set of intermediate states and relax it using gradient information from the PES. The NEB method, which is described in Section 2, is a commonly used example of such an approach that also serves as the reference baseline in this work. Another deterministic approach is the string method that represents a path as a smooth curve instead of a NEB spring chain [6]. Some of its variants are the Simplified String Method (SSM) [16] and the Growing String Method (GSM) [17]. The idea of the former is to drop the force-projection step and evolve the path under the full potential force, which makes the procedure simpler and more stable. With respect to the latter, GSM does not start iterating from the full path. It grows two fragments inward from the start and the target until they meet, which reduces the number of guesses that have to be made for the system. All the described methods depend on hyperparameters such as the number of images and the step size, together with the spring constant in the case of NEB. Therefore, their values have to be tuned accordingly for every new PES considered.

The above-mentioned deterministic methods are accurate on small systems but share several limitations. First, they require knowing the location of the endpoints in advance. Even though they might be approximated by an initial guess, this slows convergence or moves the result of the algorithms toward a wrong path. Second, the methods are computationally expensive. Usually, the potential is unknown before the experiment, so the methods query an energy model at every step, which in real molecular systems is density functional theory [18, 19] or a force field [20]. These evaluations are costly, the number of required evaluations grows with the dimensionality of the system, and the finite number of images makes the recovered path only an approximation. This makes the deterministic algorithms not applicable to every MEP finding problem, and highlights importance to search for alternative methods.

Machine learning for reaction paths. As another approach to accelerate reaction-path calculations, ML algorithms have been considered. For example, Schreiner et al. and Koistinen et al. [21, 8] replace the costly energy and force evaluations inside NEB with a learned surrogate model that recovers reaction paths much cheaper. However, this approach still relies on the iterative search and leaves other limitations of the NEB method. On the other hand, the application of other ML methods leads to the prediction of the result directly. Duan et al. [22] use a generative model to map reactants to transition states in a single pass, whereas supervised models predict transition-state structures from the reactant and product [23, 24, 25]. These predictors are fast at inference, but they have to be trained on a dataset of known transition states, which is scarce and expensive to produce. Thus, their reach is limited on new or large systems.

Reinforcement learning on potential energy surfaces. RL has become a standard approach to sequential decision problems and has recently been applied to molecular simulation. RL agents do not need a dataset of solved transition states or paths, which was a problem with the other ML algorithms. However, only a small number of works utilize RL agents to act on PESs. As an

early example of such research, Mills et al. [12] train a deep deterministic policy gradient agent to navigate the Müller-Brown surface to its global minimum. Although their reward formulation was strongly target-driven and did not focus on the full reaction-path recovery, they showed that the reward design strongly affects exploration efficiency. On the same landscape, the paper of Pal et al. [10] has a goal similar to the one of this work. In particular, they search for the lowest barrier between two minima with a reward equal to the negative energy of the next state. The agent crosses the saddle, but overestimates it and never reaches the target.

Barrett and Westermayr [26] also use RL on a PES, but the agent optimizes a different objective. It treats an entire NEB-style chain of molecular structures as the state and adjusts the atomic positions of all images step by step. With this scheme they recover minimum energy paths, and hence transition states, for a Claisen rearrangement and several S_N2 reactions. However, this is different from this work because they refined a whole path at once rather than guiding a single system from start to target. At the other extreme, Kikkawa et al. [27] use the simplest reward among all these examples. They give a fixed bonus for reaching a transition-state region and a small penalty at every other step. A deep Q-network reads the surface curvature and walks uphill from a minimum to the saddle, so the agent locates the transition state but never recovers the full path between the two minima.

Each above-mentioned study makes different reward design decisions, even though none fully justifies this choice or compares the formulations against one another. The present work closes this gap by comparing two reward functions from the studies of Pal and Mills and introducing a new reward formulation. These two studies have the most promising design and are close to the current objective, which is the reason for applying them to this thesis setup. The details for the comparison technique and the choice of algorithms are described in the next section.

4 Methodology

To investigate whether the problem of finding a MEP can be addressed using RL, two algorithms are considered: Value Iteration (VI) and Proximal Policy Optimization (PPO). Each is applied to a different representation of the same underlying environment, where the former operates on a discretized version of the Müller-Brown potential and the latter is trained in the corresponding continuous setting. In particular, the VI agent is trained on the three reward formulations to choose the one which performs the best. Each formulation is applied three times with different discount factors to test the robustness of the experiments. For further training of the PPO agent, the best reward design among those three is chosen and then applied with its best performing γ values, which finishes up the experiments. This section describes the environment in more detail, formulates the task as an RL problem, explains the reward functions under investigation, and discusses the solution methods in depth.

4.1 Markov Decision Process

The above-mentioned sequential decision problem is formalized here as a Markov Decision Process. The formulation is presented in the general case of a high-dimensional configuration space, although the experiments in this thesis are restricted to two dimensions. This restriction allows for future extensions of this work to other spaces and keeps the focus on the central object of investigation,

the reward function formulations.

A state $\boldsymbol{s} \in \mathcal{S} \subseteq \mathbb{R}^{3n}$ is a $3n$-dimensional coordinate vector $(s_1, s_2, ..., s_{3n})$ in the configuration space. An action $\boldsymbol{a} \in \mathcal{A}$ is a displacement of fixed magnitude Δ to a neighboring position along any combination of coordinates, $\mathcal{A} = \{\Delta \boldsymbol{d} : \boldsymbol{d} \in \{-1, 0, +1\}^{3n} \setminus \{\boldsymbol{0}\}\}$. With respect to an abstract two-dimensional configuration space with two coordinate values (s_1, s_2), it does not correspond to an integer n number of atoms as it was already mentioned in Section 2. Still, the same rule gives $3^2 - 1 = 8$ actions covering the horizontal, vertical, and diagonal directions. The transition function is deterministic, $\mathcal{T}(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a}) = 1$, with $\boldsymbol{s}' = \boldsymbol{s} + \boldsymbol{a}$, and the discount factor γ should be tuned based on the choice of the reward function formulation.

Two versions of the state space are considered. In the continuous version, a state is the coordinate vector of any point in it. In the discrete version, the configuration space is partitioned into a uniform grid of cells, and a state indexes one of these cells. The continuous version is used by the PPO agent and the discrete version by Value Iteration.

The remaining component of MDP is the reward function $\mathcal{R}$, which must encode the notion of following a physically meaningful reaction path. In this thesis, three reward functions are evaluated, among them being

$$\mathcal{R}^{\text{next}}(\boldsymbol{s}_t, \boldsymbol{s}_{t+1}) = -E(\boldsymbol{s}_{t+1}) \tag{7}$$

$$\mathcal{R}^{\Delta}(\boldsymbol{s}_t, \boldsymbol{s}_{t+1}) = -\big(E(\boldsymbol{s}_{t+1}) - E(\boldsymbol{s}_t)\big) = -\Delta E(\boldsymbol{s}_{t+1}, \boldsymbol{s}_t) \tag{8}$$

$$\mathcal{R}^{+}(\boldsymbol{s}_t, \boldsymbol{s}_{t+1}) = -\max\big(E(\boldsymbol{s}_{t+1}) - E(\boldsymbol{s}_t),\, 0\big) \tag{9}$$

The *next state energy reward* $\mathcal{R}^{\text{next}}$, used by Pal et al. [10], assigns a reward that is proportional to the negative energy of the next state $\boldsymbol{s}_{t+1}$. In other words, the agent is rewarded for being in lower energy regions, which might encourage the agent to reach the minima faster. Overall, the cumulative reward is the discounted sum of the energies visited along the trajectory, which depends only on which states are visited. The MEP, in contrast, minimizes the energy at every point along the path in all directions perpendicular to it, which is different from the total energy along that way.

The *energy difference reward* $\mathcal{R}^{\Delta}$, used by Mills et al. [12], is the energy difference between the two states, which measures the change in energy along the transition. In particular, uphill moves are penalized and downhill moves are rewarded by the same magnitude, so locally the agent prefers descending. Globally, however, the cumulative reward along any path from A to B is equal to $-\big(E(B) - E(A)\big)$, which depends only on the endpoints. Therefore, two paths with different barrier heights produce the same return, and the reward signal carries no information about which is preferable.

The *non-positive reward* $\mathcal{R}^{+}$, proposed in this thesis, only punishes an agent for an energy increase, where it assigns zero to downhill moves. This means that the cumulative reward depends on how much the agent climbs up and thus increases the energy along the path. In contrast to $\mathcal{R}^{\Delta}$, trajectories from A to B that cross higher barriers result in larger penalties. Therefore, the agent is encouraged to find paths with minimal uphill cost, which is closely related to minimizing the activation energy. For example, in the case of one barrier on the landscape, these two are identical. At the same time, this reward makes descending free, which makes it match the physical intuition. In other words, on an energy landscape, going up is the unlikely, rate-limiting part, while going down happens easily.

4.2 Algorithms

Value Iteration. To find the optimal policy on the discretized Müller-Brown surface, dynamic programming (DP) methods are applied first, as they provide an exact solution when the transition function and reward function are fully known. Specifically, we use Value Iteration (VI) algorithm, which computes the optimal action-value function $q^*(\boldsymbol{s}, \boldsymbol{a})$ by repeatedly applying the Bellman optimality update,

$$Q(\boldsymbol{s}, \boldsymbol{a}) = \sum_{\boldsymbol{s}'} \mathcal{T}(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a})[\mathcal{R}(\boldsymbol{s}, \boldsymbol{s}') + \gamma \max_{\boldsymbol{a}'} Q(\boldsymbol{s}', \boldsymbol{a}')], \tag{10}$$

until convergence. In this equation, $Q(\boldsymbol{s}, \boldsymbol{a})$ is the estimate of the true action-value function $q(\boldsymbol{s}, \boldsymbol{a})$ that the agent stores during iterations, and $\max_{\boldsymbol{a}'} Q(\boldsymbol{s}', \boldsymbol{a}')$ represents the estimate of the maximum expected return from the next state $\boldsymbol{s}'$ over all possible actions $\boldsymbol{a}'$. Moreover, since the transitions are deterministic, $\mathcal{T}(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a})$ has only one entry equal to 1, so the sum over $\boldsymbol{s}'$ reduces to one term.

Proximal Policy Optimization. Unlike VI, Proximal Policy Optimization (PPO) method does not require a model of the environment. It is a policy gradient actor-critic method, that is considered to be highly stable due to its clipping mechanism [14]. In particular, a plain policy gradient step can move the policy too far in a single update and destabilize training. Therefore, rather than optimizing J directly, PPO maximizes a clipped surrogate objective that limits the size of each update,

$$L^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}_t\Big[\min\Big(r_t(\theta)\,\hat{A}_t,\ \text{clip}(r_t(\theta),\,1-\epsilon,\,1+\epsilon)\,\hat{A}_t\Big)\Big]. \tag{11}$$

Here, $r_t(\theta) = \dfrac{\pi_\theta(\boldsymbol{a}_t \mid \boldsymbol{s}_t)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_t \mid \boldsymbol{s}_t)}$ is the ratio between the new and old policy probabilities of an action $\boldsymbol{a}_t$, $\hat{A}_t$ is the estimated advantage, and ϵ is the clipping range. This ratio is an importance-sampling correction that allows PPO to take several gradient steps on the same batch of transitions collected by running the policy in the environment. The old policy is the one that collected the batch, and the new policy is the one being updated. The clipping of $r_t(\theta)$ to $[1-\epsilon, 1+\epsilon]$ keeps the two policies close enough that the correction stays valid.

4.3 Müller-Brown potential

Since this thesis focuses on designing reward functions that capture the structure of the true MEP, the task is simplified to a low-dimensional setting. In this work, we use the benchmark two-dimensional Müller-Brown potential because it is frequently used to validate different methods for finding reaction pathways [11]. From now on we denote a point on the surface by its two coordinates x and y, rather than the general vector $\boldsymbol{s} \in \mathbb{R}^{3n}$, since there is no integer n to match the physical description here. The potential consists of a weighted sum of Gaussian functions,

$$E(x, y) = \sum_{k=1}^{4} A_k\, e^{a_k(x-x_k)^2 + b_k(x-x_k)(y-y_k) + c_k(y-y_k)^2},$$

where $(x, y) \in \mathbb{R}^2$ denotes a point on the surface. The values for x and y are truncated to be in range $x \in [-1.7; 1.3]$ and $y \in [-0.4; 2.1]$, accordingly. Each term in the sum represents a Gaussian contribution centered at x_k, y_k. The hyperparameter A_k controls the amplitude of the k-th Gaussian, while a_k, b_k, and c_k determine its shape and orientation.

The surface is implemented as an OpenAI Gym environment and could be seen in Figure 2. The hyperparameters are initialized according to the standard values of the original paper such that the landscape consists of three local minima that are connected via two saddle points. The exact values are depicted in Table 1. The start position, denoted by A, is located at the reactant minimum $A = (0.623, 0.028)$ and has energy $E(A) = -108.167$, while the target position $B = (-0.558, 1.442)$ lies in a lower-energy basin with energy $E(B) = -146.7$. The energies of the model are arbitrary units rather than being on any physical scale. The two saddle points on the surface, namely, at $(0.212, 0.293)$ and $(-0.822, 0.624)$ lie on the way between these two endpoints, but are still separated by another intermediate valley at $(-0.050, 0.467)$. This structure makes it a suitable test case for studying transition pathways on a landscape, while still preserving computational efficiency.

Table 1: Parameters of the four-Gaussian Müller–Brown potential.

k	A_k	a_k	b_k	c_k	x_k	y_k
1	-200	-1	0	-10	1	0
2	-100	-1	0	-10	0	0.5
3	-170	-6.5	11	-6.5	-0.5	1.5
4	15	0.7	0.6	0.7	-1	1

Each episode starts from the reactant minimum A and terminates when the agent enters a neighbourhood of radius $d_{\text{succ}} = 0.1$ around the product minimum B. There are eight possible actions in the environment, namely, right, left, up, down, up-right, up-left, down-right, down-left. Moreover, an agent cannot leave the state space, because after an action that attempts to cross the border, the agent receives a boundary penalty of -5 and stays in place. The reason for this design choice is to avoid infinite loops with 0 or positive returns. In other words, after pushing the border, there is no reason to support an agent by a positive reward, since it is an undesired behavior. And if the reward is 0, it can idle near the border for free. This behavior can then outvalue any trajectory that must pay to cross the barrier, so a negative penalty is added to make self-loops strictly worse than terminating. The penalty is set to -5, although on this PES its exact magnitude is not important. This is because reaching any boundary requires climbing much higher in energy than reaching the product minimum, making the former an undesired destination. Thus, even if the agent could reach the edge and wait there to discount future costs, the additional cost of reaching the boundary already outweighs any potential benefit. Consequently, boundary actions are still left out of the optimal trajectory, but this penalty needs to be carefully chosen on the other potentials.

The same environment is used for both algorithms, but it is represented differently depending on the method. VI uses a discretized grid, while PPO acts directly in the continuous coordinate space. Also, if during a training phase, the PPO agent does not reach the goal within $t_{\max} = 1000$ steps, the episode is truncated. However, VI has an infinite horizon while training, and both methods have $t_{\max} = 1000$ steps at the evaluation phase.

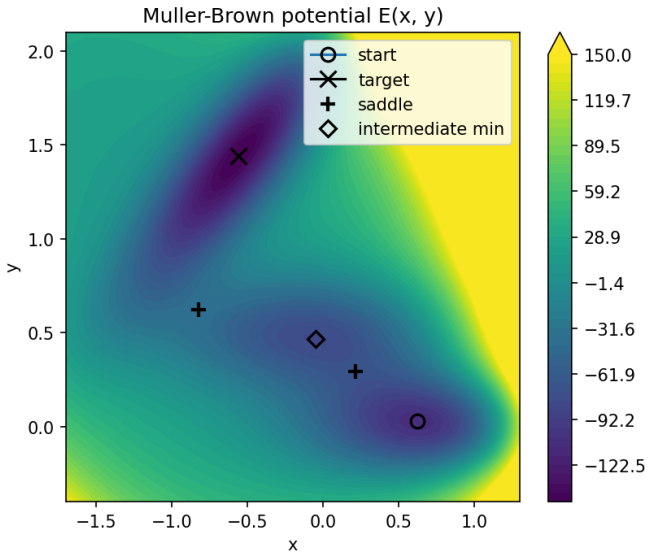


Figure 2: The Müller-Brown potential $E(x, y)$ over the continuous state space. Dark regions correspond to low-energy minima and bright regions to high-energy barriers. The circle marks the start state $A = (0.623, 0.028)$, the cross marks the target state $B = (-0.558, 1.442)$, the plus markers indicate the two saddle points, located at $(0.212, 0.293)$ and $(-0.822, 0.624)$ and the diamond marks the intermediate minimum, located at $(-0.050, 0.467)$. Among the two saddle points, the latter corresponds to the higher barrier. The agent must reach the target B by going through the first saddle point from the start A, passing by the intermediate basin, visiting the second saddle point and terminating at the end B.

4.4 Experimental set-up

The NEB reference path is computed first and serves as the baseline against which the VI and PPO agents are compared. It is obtained with the climbing-image variant on the Müller-Brown potential, using 50 images, a spring constant of 2.0, a step size of 4×10^{-3}, and at most 4500 iterations with a convergence tolerance of 5×10^{-4}. The remainder of this section gives the settings for the two agents, namely how the continuous Müller-Brown environment is discretized for VI, how PPO is trained in the continuous setting, and how the resulting policies are evaluated against the reference.

Value Iteration. VI serves as a reference for comparing the reward formulations, because it has full knowledge of the model and returns the optimal policy on the discretized MDP. Therefore, it is the first implemented algorithm for the experiments.

To run VI, the continuous state space is discretized into a uniform 20×20 grid, and the reward functions take the center positions of the grid cells as input. The Q-values are initialized to zero and updated via synchronous Bellman backups until $\max_{\boldsymbol{s},\boldsymbol{a}} |\Delta Q(\boldsymbol{s}, \boldsymbol{a})| < 10^{-2}$. The three reward functions from Equations 7, 8, 9 are evaluated under three discount factors $\gamma \in \{0.99, 0.999, 1\}$ to test how sensitive the resulting policy is to this hyperparameter. This gives nine configurations, one for each pair of reward and discount value.

The assessment of each reward function includes comparing trajectories and activation energies to the NEB baseline. With respect to the trajectories, they are produced by nine configurations of

the reward-discount pairs by performing a roll-out from the start state until environment termination or truncation. The best γ values for each reward design are then chosen by visual shape assessment to resemble the baseline. Afterwards, the energy profiles are plotted for each reward function under this choice of γ. The activation barriers are quantitatively evaluated for matching the NEB reference, and the best pair is carried forward to the PPO experiments.

Another way to compare the trajectory of the agent and the reference is through their distance in state space, such as the mean squared distance between the two paths after normalizing them to a common length. However, the quantity of physical interest is the activation barrier, not the geometric measure of similarity. A geometric distance quantifies a different property from the activation barrier, and there is no unique definition of such a distance. Different alignment procedures and distance measures (e.g., mean squared distance, Hausdorff distance [28], or Fréchet distance [29]) emphasize different aspects of the trajectory and may lead to different conclusions. Since the likelihood of the transition is mostly determined by the activation height, the evaluation focuses on the energy profile and the resulting barrier.

Proximal Policy Optimization. PPO is trained with a CleanRL-style implementation [30] on the continuous state space, using the reward formulation chosen from the VI experiments. Both agents use the same eight discrete actions, namely, the orthogonal and diagonal displacements of magnitude 0.1. The difference between them is the state representation, where PPO does not snap each position to its cell center and acts on the continuous state without confining with the grid. With respect to the architecture, the actor and critic are two-layer MLPs with tanh activations. The actor uses hidden layers of size 256, and the critic uses hidden layers of size 128 and 256. Networks are initialized with orthogonal weights following the CleanRL defaults. The training hyperparameters are listed in Table 2.

Table 2: PPO training hyperparameters.

Parameter	Value
Total timesteps	10^8
Parallel environments	4
Rollout length per environment	700
Optimiser	Adam ($\epsilon = 10^{-5}$)
Learning rate	2.5×10^{-4} (linearly annealed)
Discount factor γ	1.0
GAE λ	0.95
Minibatches per update	4
Update epochs K	4
Clip coefficient	0.2
Value-loss clipping	enabled
Value-loss coefficient	0.7
Entropy coefficient	0.05
Advantage normalisation	enabled
Max gradient norm	0.5

From the table, the advantage estimates used by PPO are computed with generalized advantage

estimation (GAE) [31] using $\lambda = 0.95$. Therefore, since the true advantage $A^{\pi_\theta} = q^{\pi_\theta} - v^{\pi_\theta}$ is not available, GAE approximates it from temporal-difference residuals,

$$\delta_t = \mathcal{R}(\boldsymbol{s}_t, \boldsymbol{s}_{t+1}) + \gamma V^{\pi_\theta}(\boldsymbol{s}_{t+1}) - V^{\pi_\theta}(\boldsymbol{s}_t),$$

which are accumulated as

$$\hat{A}_t^{\pi_\theta} = \sum_{l \geq 0} (\gamma\lambda)^l \delta_{t+l}.$$

The parameter λ here controls the degree of bootstrapping from the critic. In this thesis, $\lambda = 0.95$ is used as a CleanRL default that keeps the estimate close to the observed returns.

The experiments use 5 seeds to train the agents, namely, $\{0, 1, 2, 3, 4\}$, chosen consecutively from 0 to avoid cherry-picking. This task is difficult from the exploration point of view, because the low-cost region that the agent seeks lies on the other side of the barrier. To get there by crossing the saddle from the start, an agent must incur a sequence of costly steps first, and the entropy bonus alone did not push the agent through in early experiments. To guarantee coverage of the space, we instead start each episode from a uniformly random position on the landscape with probability 75%, and from the reactant minimum A otherwise. This randomization places some episodes directly beyond the barrier, so the value estimates there can propagate back and inform the policy near the start. Since this method spends much of the sampling in regions that lie away from the path, the training budget amounts to 100M timesteps to still cover the relevant region.

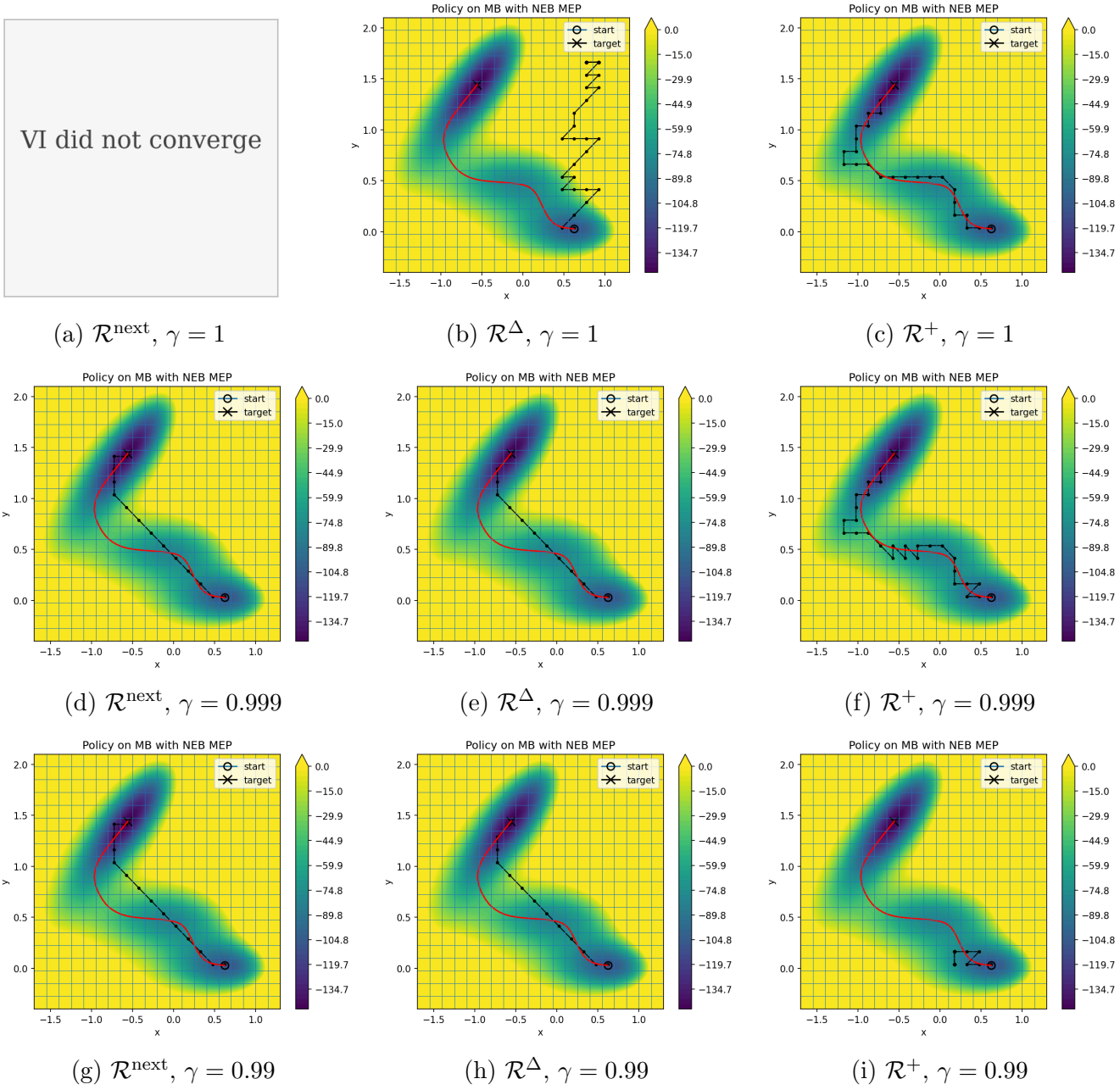
The trained PPO policies are assessed in the same way as the VI policies, by comparing their trajectories and activation energies to the NEB baseline. With respect to the trajectories, the agents are evaluated at six checkpoints, at 1%, 10%, 50%, 70%, 90%, and 100% of the total training steps. At each checkpoint the model is saved and a deterministic roll-out is run from the start state A, taking the most likely action at each step rather than sampling. As a result, it becomes possible to visualize the trajectories of each seed in one plot. With regards to the activation energies, the energy profile of the final trajectory of each seed is drawn and compared against the NEB reference. Similar to VI, the values of the energy barriers are also displayed to quantitatively assess the performance.

In addition, the episodic returns of the training process are collected and reported as learning curves, which makes the analysis of the training progress possible. Two plots are produced, with the return of each completed episode on the x-axis and the timestep at which the episode ends on the y-axis. As there are 5 training seeds, the run for each seed is smoothed with Savitzky–Golay filter before plotting. The smoothing window is 250001, while the polynomial degree is 1. The first plot shows the mean return and its standard deviation across the five seeds, whereas the second plot shows the learning curve of each seed on its own. Due to the fact that each episode terminates at different timesteps across the seeds, the results for the first plot are interpolated onto a common grid. Its length was determined by the seed with the largest number of episodes.

The full implementation, including the custom OpenAI Gym environment, is available at https://github.com/Sasha580/MEPRL_thesis.git.

5 Results

The results of the VI agents are displayed in Figures 3, 4 and 5. Figure 3 shows the trajectories of all agents, Figure 4 zooms in the Q-values of the VI agent with $\mathcal{R}^+$ under $\gamma = 1$, and finally,



(a) $\mathcal{R}^{\text{next}}$, $\gamma = 1$ (b) $\mathcal{R}^{\Delta}$, $\gamma = 1$ (c) $\mathcal{R}^{+}$, $\gamma = 1$

(d) $\mathcal{R}^{\text{next}}$, $\gamma = 0.999$ (e) $\mathcal{R}^{\Delta}$, $\gamma = 0.999$ (f) $\mathcal{R}^{+}$, $\gamma = 0.999$

(g) $\mathcal{R}^{\text{next}}$, $\gamma = 0.99$ (h) $\mathcal{R}^{\Delta}$, $\gamma = 0.99$ (i) $\mathcal{R}^{+}$, $\gamma = 0.99$

Figure 3: Optimal policies obtained from VI for the three reward functions (columns) at $\gamma = 1$ (top row), $\gamma = 0.999$ (middle row), and $\gamma = 0.99$ (bottom row). Only the custom reward $\mathcal{R}^{+}$ recovers a path resembling the NEB reference, and only when γ is close to 1. The other two reward functions fail to produce a meaningful policy at $\gamma = 1$ and collapse to a shortcut path at $\gamma < 1$. When γ is decreased, the agent is biased toward reaching low-energy regions within a shorter horizon, which encourages heuristic policies that rely less on the underlying energy landscape.

Figure 5 presents the energy profiles of the final resulting policies under $\gamma = 1$ for $\mathcal{R}^+$ and under $\gamma = 0.999$ for the other rewards. In-depth analysis is presented in the following sections.

5.1 VI.

Trajectories. Figure 3 shows the optimal policies obtained from VI for the three reward functions at $\gamma \in \{1, 0.999, 0.99\}$. The columns of the figure correspond to each reward design in question, whereas the rows correspond to decreasing values of γ. As can be seen in the graphs, the policies depend on the discount factor in two opposite ways. In particular, the first two rewards $\mathcal{R}^{\text{next}}$ and $\mathcal{R}^{\Delta}$ both fail to find the MEP at $\gamma = 1$, and could reach the goal only under the condition of $\gamma < 1$. Discounting biases the agent to obtain low-energy rewards within a shorter horizon, which leads to heuristic shortcuts instead of following the underlying energy landscape. In contrast, the custom reward $\mathcal{R}^+$ at $\gamma = 1$ produces a policy that closely resembles the NEB baseline, but becomes highly sensitive to small changes in the discount factor. The details on the result of each reward are discussed further in the following paragraphs.

$\mathcal{R}^{\text{next}}$. The results of the first reward function are depicted in Figures 3a, 3g and 3d. Figure 3a with $\gamma = 1$ does not contain a trajectory, unlike the other panels, because such a configuration of the algorithm does not converge to a policy. This is a consequence of the intrinsic structure of the reward function, where the reward depends only on the energy of the next state. In other words, most transitions on the landscape produce positive rewards because the majority of the states on the grid have negative energy. As a result, the surface contains loops with positive return, and under the infinite episode horizon of the VI objective, value function diverges at $\gamma = 1$. To deal with this issue, discounting with $\gamma < 1$ makes these returns finite, but this setting still has the downsides discussed below.

When the γ values are decreased slightly below one, as for $\gamma = 0.999$ and $\gamma = 0.99$, shown in Figures 3d and 3g, respectively, the value function converges. Subsequently, in both cases the optimal policy becomes the direct path from A to B which deviates significantly from the true MEP. This occurs because the discount factor vanishes the contribution of long-term gains and therefore biases the agent toward shorter trajectories. Since the final minimum corresponds to the lower-energy basin, the policy attempts to reach it more quickly.

$\mathcal{R}^{\Delta}$. The results for the second reward function $\mathcal{R}^{\Delta}$ are displayed in Figures 3b, 3e and 3h. At $\gamma = 1$ in Figure 3b, the resulting trajectory is oscillating left-right, while making overall upward progress. The path does not reach the target, while being settled into a loop instead. The issue is again a direct consequence of how $\mathcal{R}^{\Delta}$ is constructed. In other words, any paths connecting two endpoints A and B are indistinguishable from each other, with the cumulative reward being equal to $-\Delta E = E(A) - E(B) > 0$ for all of them. The choice between all these paths is determined by tie-breaking among equally good actions, and any preference between routes must come from discounting. Consequently, under $\gamma = 1$ a round trip between two neighbouring cells gives zero reward, so reaching B and staying in place give the same return. The policy settles on a loop and does not reach the target.

At $\gamma < 1$, shown in Figures 3e and 3h, the trajectories resemble the ones of $\mathcal{R}^{\text{next}}$ and take the same shortcut between A and B rather than following the MEP through the two saddle points. The only difference in the paths with $\mathcal{R}^{\text{next}}$ is the absence of a one-step detour near the target position

B, which is likely an artifact of the discretization. Compared to $\gamma = 1$, the path independence no longer holds, where the cumulative reward decomposes as

$$G_{1:N} = \sum_{t=1}^{N} \gamma^{t-1} \mathcal{R}_t^{\Delta}(s_t) = E(A) - \gamma^{N-1} E(B) - (1-\gamma) \sum_{t=1}^{N-1} \gamma^{t-1} E(s_t),$$

with N being the total number of steps to the goal. Since $E(B) < 0$, the terminal contribution $-\gamma^{N-1}E(B)$ is larger when N is smaller, because less of B's reward is discounted away. The optimal policy at $\gamma < 1$ therefore prefers shorter trajectories and collapses to the same shortcut produced by $\mathcal{R}^{\text{next}}$, despite the different mechanism underlying the two rewards.

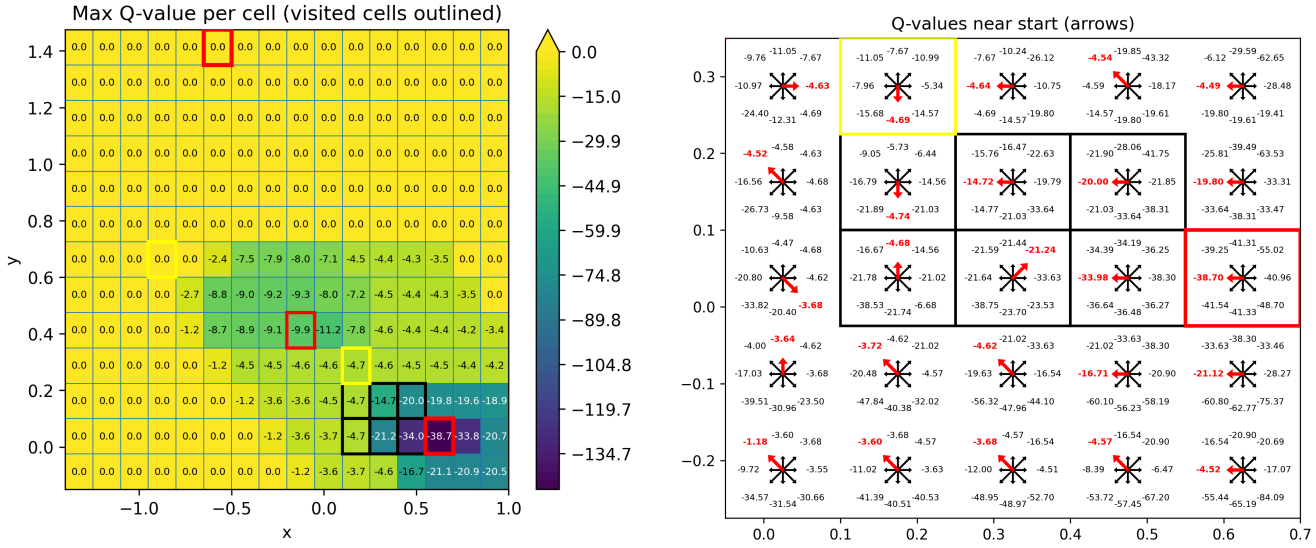
$\mathcal{R}^{+}$. The learned policy for the custom reward design can be observed in Figures 3c, 3f and 3i. At $\gamma = 1$, displayed in Figure 3c, VI finds a trajectory that closely matches the NEB reference by visiting both saddle points and reaching the goal. This is the only reward function that works with $\gamma = 1$ and is structurally designed to minimize the same quantity as MEP. In particular, along any trajectory, the cumulative reward of $\mathcal{R}^{+}$ equals the negative sum of all the encountered uphill energy contributions, which is preserved unchanged at $\gamma = 1$.

At $\gamma = 0.999$, shown in Figure 3f, the policy still reaches the target but the trajectory now starts fluctuating along the path, and differs from the smoother result at $\gamma = 1$. This is likely a consequence of grid discretisation, and a finer grid would probably produce a smoother trajectory even when $\gamma < 1$. Interestingly, if the value of γ is decreased even further, the policy no longer reaches the target B, which can be observed in Figure 3i. At $\gamma = 0.99$, after a few zigzag steps from A, the agent settles into a cycle between two neighboring cells in the starting basin. This suggests that this value of γ discounts future rewards too aggressively, so that the one-time cost of crossing the barrier looks worse than the small zigzag steps made inside the basin.

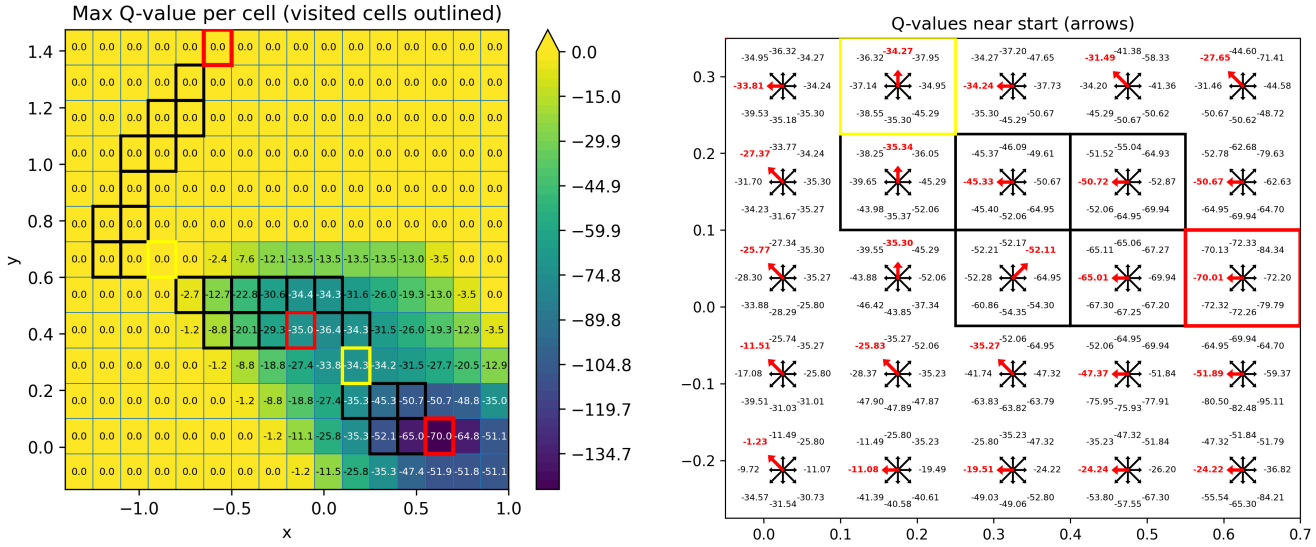
Although the undiscounted setting provides useful insight into the behavior of the reward functions, the discounted setting is of greater practical relevance. In deep reinforcement learning, such discount factors ensure stable learning of the value function estimates. Furthermore, when the terminal state is not known in advance, the planning horizon is limited through $\gamma < 1$ rather than explicit episode termination. Therefore, the following analysis focuses on the discounted setting in greater detail.

Q-values per cell. The difference in behaviours at $\gamma = 0.99$ and $\gamma = 0.999$ is most visible in the Q-values learned by VI. This is depicted in Figure 4, where each subfigure contains two panels. The left panel shows the maximum Q-value per cell on the discretised landscape. The right panel zooms into the region around the start state A and displays the Q-values of all eight actions per cell. Comparing the two subfigures, the Q-values grow noticeably in magnitude as γ increases. For example, the start cell A has a discounted return of approximately -70 under $\gamma = 0.999$, whereas that value is around -39 for $\gamma = 0.99$.

The weak propagation at lower γ also has a visible effect on the policy behavior. For instance, at $\gamma = 0.99$ in Figure 4a left, once the agent starts climbing out of the initial basin A, it settles into a loop directly before the yellow saddle-point cell. Therefore, the discounted future return past the saddle is too small to outweigh the immediate cost of the climb. From the right panel of the same figure, this difference in the Q-values is highlighted. In particular, at that cell, the Q-values for the climbing action and its reverse differ by only one unit in favor of the reverse, which is enough to



(a) $\mathcal{R}^+$, $\gamma = 0.99$

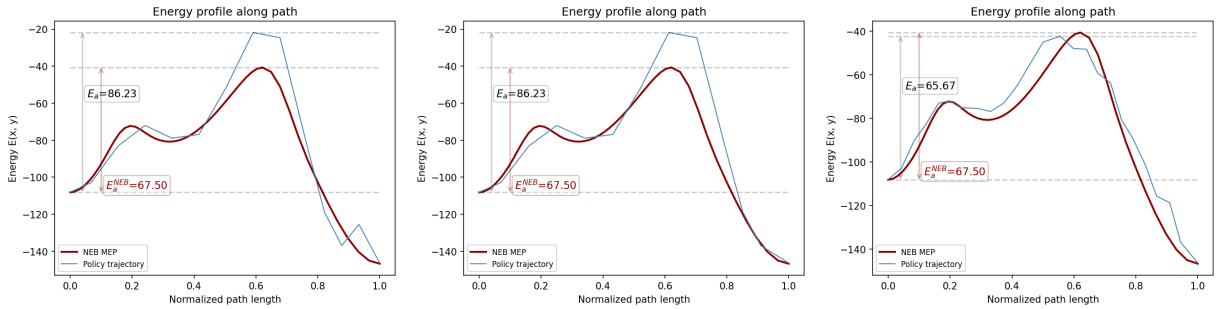


(b) $\mathcal{R}^+$, $\gamma = 0.999$

Figure 4: Q-values learned by value iteration under $\mathcal{R}^+$ at two discount factors. The left panel of each row shows the maximum Q-value per cell across the discretised landscape, where brighter cells have higher value. The right panel zooms into the region around the starting state A and shows the Q-value of each of the eight available actions per cell, with the greedy action drawn as a red arrow. In both panels the start, target and intermediate basins are outlined in red, the cells visited by the greedy policy in black, and the two saddle points in yellow.

lock the policy into the loop. As it is seen from the Figure 4a the cumulative uphill cost along the trajectory is on the order of tens of units. At $\gamma = 0.99$ discounting shrinks this future cost so much that crossing the barrier looks worse than cycling near the start.

At $\gamma = 0.999$ the agent can escape from the initial and intermediate basins. In Figure 4b left, the agent successfully passes through both saddle points, and starts gaining zero rewards by going downhill after the second one. In comparison with the $\gamma = 0.99$ setting, Figure 4b (right) shows the flip in direction near the first saddle point, where a climbing action accounts for 0.03 more value than its reverse. This marginal sufficiency of $\gamma = 0.999$ here suggests that once landscapes have more basins or are discretized finer, hyperparameter tuning becomes extremely difficult there. In those settings, trajectories become longer and the per-step uphill costs shrink, so the same discount factors propagate even less. As a result, for $\mathcal{R}^+$, setting $\gamma = 1$ becomes the safer default. However, it is important to highlight again that training without discounting in the deep setting may introduce its own stability difficulties.



(a) $\mathcal{R}^{\text{next}}$, $\gamma = 0.999$ (b) $\mathcal{R}^{\Delta}$, $\gamma = 0.999$ (c) $\mathcal{R}^{+}$, $\gamma = 1$

Figure 5: Energy along the policy trajectory against normalized path length, with the NEB MEP overlaid as reference. Only (c) $\mathcal{R}^+$ produces a policy that follows the NEB profile, where (a) $\mathcal{R}^{\text{next}}$ and (b) $\mathcal{R}^{\Delta}$ crosses a peak much higher than the NEB barrier.

	NEB	$\mathcal{R}^{\text{next}}$	$\mathcal{R}^{\Delta}$	$\mathcal{R}^{+}$
γ	—	0.999	0.999	1
E_a	67.50	86.23	86.23	65.67
$E_a - E_a^{\text{NEB}}$	—	+18.73	+18.73	−1.83

Table 3: Activation energy E_a of the VI policy under each reward function, together with the discount factor used, compared to the NEB reference. The first two rewards cross a barrier roughly 19 units above the reference, while $\mathcal{R}^+$ matches it within rounding.

Energy profiles. Figure 5 shows the energy encountered along the policy trajectory for the three reward functions in question. The curves were plotted against the NEB baseline and were normalized against the path length. As it is seen in Figures 5a and 5b, the first two reward functions do not recover the true MEP because their resulting paths cross a peak of high energy instead of the lower saddle point. In contrast, the custom reward formulation in Figure 5c follows the NEB

profile and reaches near the same activation barrier as the reference. These results are summarized in Table 3, which presents the activation energies of all three configurations. The custom reward function achieves the peak that is 1.83 lower than the NEB reference, whereas the other reward formulations overshoot the reference by 18.73.

Under $\mathcal{R}^{\text{next}}$ and $\mathcal{R}^{\Delta}$ at $\gamma = 0.999$, the policies cross a peak of near -22, around 18 units above the NEB barrier. They both share similar patterns in the activation energy because the final trajectory of two policies produced almost identical plots as shown in Figures 3e and 3d. The only difference was in the low rise in energy near the target B for $\mathcal{R}^{\text{next}}$, where the agent makes a one-step detour to the goal.

Under $\mathcal{R}^{+}$ at $\gamma = 1$, displayed in Figure 5c, the policy follows the NEB profile. The trajectory crosses both saddles, dips through the intermediate basin near path length 0.3, and reaches a peak of near -42, within a couple of units of the NEB MEP barrier. It is important to note that the slight offset at the intermediate minima and saddles is likely a discretization effect from the step size $\delta = 0.1$ rather than a policy failure.

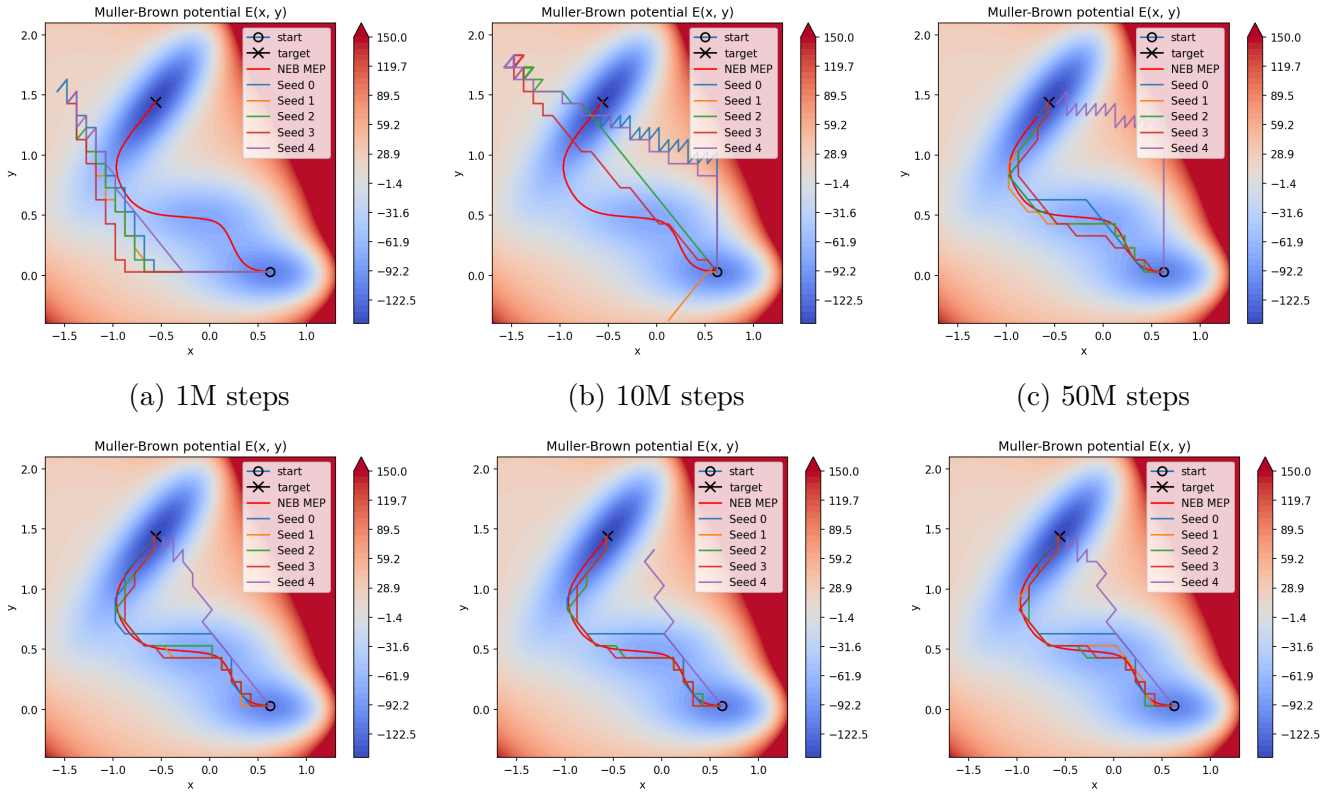
In conclusion, $\mathcal{R}^{\text{next}}$ and $\mathcal{R}^{\Delta}$ require $\gamma < 1$ to produce convergent solutions. This means that the resulting behavior is highly dependent on a discounting factor rather than on the reward formulation itself. Under $\gamma < 1$, both rewards prefer shorter trajectories between A and B instead of the true MEP, while at $\gamma = 1$ they fail to produce a meaningful solution altogether. In contrast, $\mathcal{R}^{+}$ remains well-defined at $\gamma = 1$, where the resulting policy closely follows the NEB reference. Although even small discounting weakens the propagation of the full barrier cost and can destabilize the policy, $\mathcal{R}^{+}$ is the only reward formulation suitable for the PPO experiments in the next section.

5.2 PPO

The progress of a PPO training over 100 million timesteps is displayed in Figure 6 and Figure 7 with the resulting trajectories and learning curves, respectively. Additionally, Figure 8 presents the energy profiles of the final policies evaluated after the last timestep. The runs use $\mathcal{R}^{+}$ with $\gamma = 1$, and this configuration was repeated across five seeds.

Trajectories. Figure 6 shows the greedy trajectories of the five seeds at six checkpoints during training. After 1M and 10M steps, all trajectories remain unstable and spread across different regions of the state space. The picture begins to change around 50M steps, when four of the five seeds start to follow the NEB MEP through the saddle region. From 70M steps onwards seeds 1, 2 and 3 settle into trajectories close to the NEB reference, with little additional change in shape until the end of the training. The only deterioration occurs with seed 0, which goes slightly off the intermediate basin, but still remains quite close to the baseline. In contrast, the fifth seed, shown in purple, takes a shortcut from A to B and therefore, passes through high-energy states towards the goal. Moreover, even at 90M steps this seed has not yet reached the target B, and the policy does not converge to the MEP within the training budget.

The results of the final seed 4 highlight the exploration difficulty of the task in question. Even with the full budget, the exploring starts did not place enough episodes in the region beyond the barrier for this seed, so it never learned the low path and kept to the shortcut instead. A longer training horizon for this seed might have improved the final trajectory.



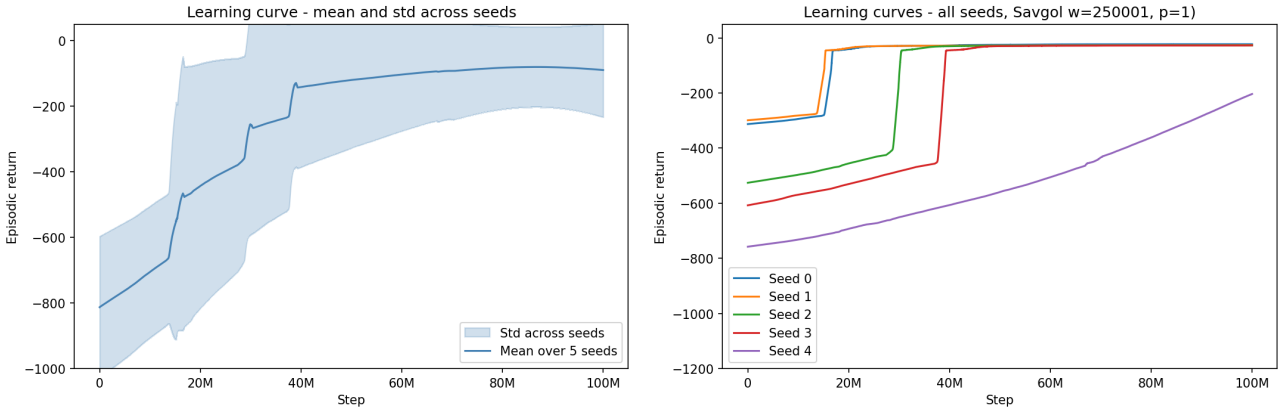
(a) 1M steps (b) 10M steps (c) 50M steps

(d) 70M steps (e) 90M steps (f) 100M steps

Figure 6: Policy trajectories from five PPO seeds at six checkpoints during training under $\mathcal{R}^+$ with $\gamma = 1$. The NEB MEP is overlaid as reference. By 100M steps four of the five seeds approximate the NEB profile, while one seed continues to fail to reach the target basin.

Learning curves. Figure 7b displays the learning curves of the PPO agent within the given time frame. The left panel 7a presents the averaged result of 5 seeds, together with its standard deviation, while the right panel 7b plots all seeds separately. As it is seen in Figure 7b, four of the five seeds reached near-zero return between roughly 15M and 40M steps and then stayed stable until the end of training. The fifth seed improved slowly across the full budget and remained near -200 in the end. These results are consistent with the results of the previous analysis, where only Seed 4 did not converge within 100M steps. From Figure 7a, the wide standard deviation reflects this spread in convergence time, since each converged seed followed a sharp step to near-zero return. The last seed never reaches the final gains of the other agents, but continues to trend upward, again suggesting a longer training budget to achieve convergence.

Energy profiles. Figure 8 presents the curves for five PPO seeds with the energies gained along the final policy trajectories. Table 4 summarizes the results of each seed and compares the activation barrier heights with the VI agents. The PPO agent recovered the activation energy in four of the five seeds, with a much smaller gap to the NEB reference than the VI agent. In other words, the four successful seeds reached 67.37 against the reference 67.50, a gap of only 0.13, compared to the 1.83 gap of the VI policy. Seed 4 was the exception, crossing at roughly twice the barrier height.



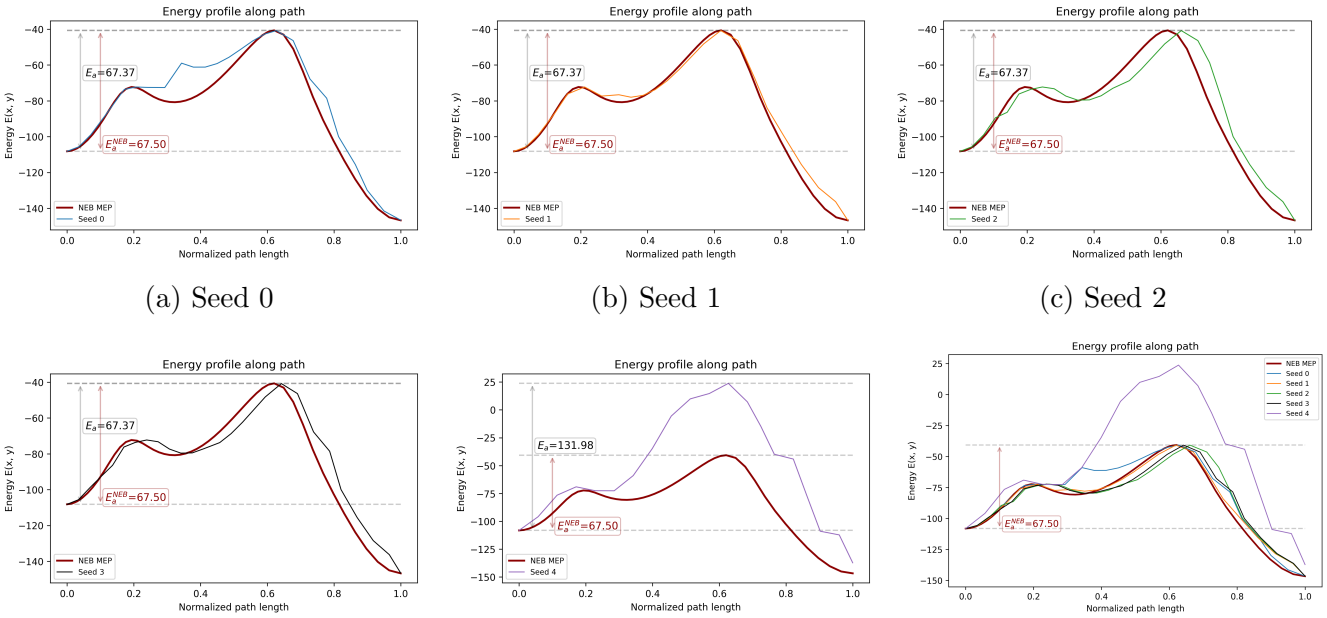
(a) (b)

Figure 7: Episodic return across five PPO seeds during 100M training steps under $\mathcal{R}^+$ with $\gamma = 1$. (a) shows the mean and standard deviation of the return across the five seeds. Each seed is smoothed with a Savitzky-Golay filter of window 250001 and polynomial order 1, and then interpolated onto a common grid whose number of points equals the largest number of recorded returns across the five seeds. The solid line shows the mean of these results and the shaded band shows the standard deviation with 4 degree of freedom. (b) shows the same smoothing applied to each seed separately. Solid lines show the smoothed curves. Four seeds transitioned sharply to near-zero return between roughly 15M and 40M steps, while seed 4 improved only gradually and remained near -200 by the end of training.

Overall, the curves of the energy profiles are smoother than the ones of the VI agent, because the PPO agent can step into regions which are not allowed to VI due to the discretization of the surface. Even more fine grained action spaces could thus produce even better results.

Looking deeper at the grid of Figure 8 reveals the energy profile of each seed. All four successful seeds recovered the same activation energy of 67.37, but their curves matched the NEB reference to different degrees. Seed 1 produced the closest curve, matching the baseline almost exactly on the uphill segments and deviating only slightly on the downhill ones. This could be compared to seeds 2 and 3 that drifted more, both before and after the peak. These deviations have two sources. First, the fixed step size $\delta = 0.1$ forces the agent to follow the path through discrete jumps. As a result, the trajectory only approximates the smooth NEB curve and the apparent peak position shifts slightly whenever a seed reaches it in a different number of steps. Second, the reward leaves the exact path underdetermined under $\gamma = 1$. In other words, since $\mathcal{R}^+$ assigns zero to downhill moves and the return equals the negative of the total uphill cost, any two paths with the same cumulative ascent give the same return. Therefore, once the barrier is crossed, the agent does not distinguish between the ways to go down. Consequently, the activation energy that depends only on the highest saddle stays at 67.37 across all four successful seeds, because they all cross the barrier.

The artifact of the reward function formulation is especially highlighted in Figure 8a. After crossing the saddle point, the agent did not go downhill to the intermediate basin, but went up and crossed it via a much higher region. This leads to a peak in energy between the two saddles. Apparently, even though the learning curve of this seed started gaining high stable rewards the earliest, the resulting converged policy is to make a detour in the environment. To investigate



(a) Seed 0 (b) Seed 1 (c) Seed 2

(d) Seed 3 (e) Seed 4 (f) All seeds overlaid

Figure 8: Energy along the final policy trajectory of each PPO seed, plotted against normalized path length and compared to the NEB MEP. Each panel reports the activation energy E_a measured from the policy trajectory and the reference value $E_a^{\mathrm{NEB}} = 67.50$. Seeds 0 through 3 all reach $E_a = 67.37$, while seed 4 crosses at $E_a = 131.98$, roughly double the reference. Panel (f) overlays all five trajectories for direct comparison. Compared to the VI results, the curves are smoother due to the more fine grained action space.

Method	Configuration	E_a	$E_a - E_a^{\mathrm{NEB}}$
NEB	reference	67.50	—
VI	$\mathcal{R}^{\mathrm{next}}$, $\gamma = 0.999$	86.23	+18.73
	$\mathcal{R}^{\Delta}$, $\gamma = 0.999$	86.23	+18.73
	$\mathcal{R}^{+}$, $\gamma = 1$	65.67	−1.83
PPO	Seed 0	67.37	−0.13
	Seed 1	67.37	−0.13
	Seed 2	67.37	−0.13
	Seed 3	67.37	−0.13
	Seed 4	131.98	+64.48

Table 4: Activation energy E_a for the PPO policies, each row per seed, and the VI policies, one per reward function. Five seeds of PPO are all trained with $\mathcal{R}^{+}$ at $\gamma = 1$. The results are compared to the NEB reference $E_a^{\mathrm{NEB}} = 67.50$. Under VI, only $\mathcal{R}^{+}$ recovers the reference barrier. Under PPO with $\mathcal{R}^{+}$, four of the five seeds recover it while seed 4 fails.

whether it is indeed the most optimal solution for this reward function in the continuous space, the cumulative rewards for all the seeds and VI agent are calculated and depicted in the Table 5.

Method	Configuration	G
VI	$\mathcal{R}^+$, $\gamma = 1$	-70.52
PPO	Seed 0	-69.91
	Seed 1	-73.82
	Seed 2	-74.70
	Seed 3	-74.70
	Seed 4	-135.53

Table 5: Rounded cumulative return G under $\mathcal{R}^+$ and $\gamma = 1$ for the VI and PPO policies. The magnitude $|G|$ is the total uphill cost along the trajectory. The cumulative reward of Seed 0 is the highest and thus, the most optimal, where the results of VI appeared to be lower due to discretization. The other seeds did not converge to the most optimal policy, although the suboptimal policies of Seeds 1, 2 and 3 showed the desired trajectory.

Returns of the agents. As it is seen in the table, Seed 0 found the cheapest path from A to B under the studied reward formulation. In particular, the return of this agent accounted for -69.91, whereas the closest gap is around 0.61 achieved by the VI agent. This suggests that although the dynamic programming agent visited the middle valley and thus was similar to the NEB baseline there, this happened mostly due to the small discretization size. Therefore, a more fine-grained landscape would cause an agent to cut the basin instead of falling into it. The results of the trained VI agent on the grid of size 40 support this claim, which can be found in A.

Taken together, the PPO experiments showed that $\mathcal{R}^+$ at $\gamma = 1$ recovered the activation barrier in four of the five seeds, but those four did not settle on one path. Three of them converged to the valley route that follows the NEB reference, while seed 0 alone found the cheaper route that cuts the basin. Importantly, the valley route is the one that matches the physical MEP, and therefore the optimum of $\mathcal{R}^+$ and the physical target do not coincide. This point is discussed further in the next section.

6 Discussion

This work compared three reward formulations for guiding an RL agent across the PES from minimum A to the lower minimum B. Only the custom reward $\mathcal{R}^+$ recovered the transition states of the reference MEP, and only with $\gamma = 0.999$ and $\gamma = 1$. In contrast, $\mathcal{R}^\Delta$ and $\mathcal{R}^{next}$ required discounting even to converge and produced a shortcut as a final trajectory. This indicates that $\mathcal{R}^+$ encodes the path-recovery objective more directly than the alternatives, though it is far from perfect and carries several limitations that are examined below.

First of all, $\mathcal{R}^+$ is highly sensitive to the values of a discount factor. In particular, already at $\gamma = 0.99$ on the 20×20 grid the agent cycles near the start state rather than crossing the barrier. This behavior is likely to become worse as the resolution increases, because the barrier cost is then

spread over a greater number of steps. Therefore, preserving the correct behavior on a finer grid would require γ very close to 1, which is extremely difficult and perhaps unfeasible to tune. In practice, however, $\gamma < 1$ is exactly what a realistic setting requires, because the terminal state of the system is usually not known in advance. The other two reward functions do not solve this issue either, being unable to recover the MEP and the transition states at any γ.

Future work could examine the application of $\mathcal{R}^+$ to the problems with infinite horizons, since this thesis focused on the episodic setting, where the terminal state is known and each run ends at the target B. The reason for such a set-up was ideal, isolated conditions to test the reward designs explicitly, where the goal of finding the path is stated clearly. Nevertheless, the question of whether $\mathcal{R}^+$ recovers the same path once the terminal state is unknown remains open, and the sensitivity to γ shown above suggests that it would be a very difficult task. $\mathcal{R}^\Delta$ and $\mathcal{R}^{next}$ rely heavily on discounting, but their objective is broken with respect to finding the MEP, which eliminates the need to test them further.

Another limitation of $\mathcal{R}^+$ comes from its objective itself. This reward minimizes cumulative ascent cost, so the correctness of the resulting path depends on the structure of the landscape. The first example is a route with lower total ascent but a higher single barrier. If a surface has another path made of more but lower barriers, its total ascent can still be larger. $\mathcal{R}^+$ then prefers the first route, even though its activation energy would be higher than the one of the MEP. The second example was evident in the experiments and is more general. If the environment is made of more than one saddle point, the agent might not visit the basin between them. This is supported by the results in the Appendix A with 40×40 grid and 0.075 action size for VI, and the observation of the PPO with the action size 0.1 and Seed 0. Apparently, the agent would pay much more if it had to climb up from the valley, which again does not coincide with the ideal behavior of the true MEP.

The above-mentioned examples disappear only when a single saddle separates the two minima. This suggests a decomposition of the task into per-barrier sub-problems. Each episode would then contain one barrier, and each sub-problem would terminate at the next minimum, where the following episode starts. However, this still raises difficulties, such as finding the intermediate minima and preventing the agent from returning to a saddle that it has already crossed.

Nevertheless, having only one saddle does not remove all the issues with the reward function. In any setting, $\mathcal{R}^+$ still says nothing about how the agent descends. In particular, a direct transition to the next minimum and a long, winding path down the hill have the same cost. This was already evident in the experiments with different seeds for the PPO agent, where the descents were slightly longer compared to the baseline. This issue might be more apparent on another landscape with more spread saddle regions. In that case, each simulation step would waste computation on the moves that do not add anything, which might also become a real problem in the infinite-horizon setting. There, nothing forces the agent to settle, and thus the agent is unlikely to find the exact valley.

Ideally, the agent should be guided to walk downhill along the gradient after crossing a saddle, so the descent is efficient and the agent settles at the minimum. The direction of steepest descent gives a signal that is local, and it points the agent along the same path that the MEP follows on its way down. However, it is still quite difficult to force this behavior by purely a reward function because it might work against the ascent cost. The total reward becomes harder to balance, since the same term that helps the descent after the saddle makes the climb toward it less attractive.

The reward signal should also be built from a change in energy rather than its absolute value. A good example of why is given by the problem with $\mathcal{R}^{next}$ reward. Its feedback is relative to the

absolute zero of the landscape, and thus, shifting that zero also shifts the policy. Such a shift does not change the MEP, though, which should be reflected by a good reward function. In contrast, if a reward is built on the energy difference, such as the downhill gradient, this avoids measuring the depth of the states, and therefore, makes the final performance safer.

Several limitations concern the experimental set-up of this study. The experiments were performed only on one version of the Müller-Brown potential, with the fixed endpoints, which might hide some of the other weaknesses of the reward design. First, the target B is always the lower minimum and the start A is the higher one. However, real molecular systems can have the reverse arrangement, and therefore, reversing A and B requires additional testing. Since this thesis focused on comparing the rewards, this arrangement was fixed.

Second, the easiest experiment to apply first is to retrain the agents on other shapes of the Müller-Brown potential. The four Gaussians can be adjusted to move the minima, change their depths, and raise or lower the barriers. This stays within the analytical 2D benchmark but varies the geometry that the reward has to handle. It would show whether the conclusions about the three rewards hold across landscapes or depend on the fixed arrangement that was used here.

Third, although the Müller-Brown potential is the canonical benchmark, it is a potential energy surface and depends only on the position of the agent. Instead, the free energy also accounts for entropy, and this contribution grows with temperature. Entropic effects strongly shape the free-energy landscape of real systems with solvent or macromolecular environments, so this landscape differs from the bare potential surfaces [32]. Therefore, additional experiments under thermal conditions should be done to check the robustness of the functions to the change in barriers and minima.

Fourth, the currently used surface is low-dimensional. On a two-dimensional surface the state space is small and the path is short, so the benchmark does not test whether RL holds any advantage over NEB in harder cases. A natural next step is to raise the dimensionality and even move to true chemical systems, then apply RL approaches there. Moreover, a further step is to train an RL method on a set of different surfaces and test whether it transfers to an unseen one. This would show whether a trained policy generalizes across instances, which NEB cannot do because it reruns from scratch for each new system.

Fifth, the budget of the PPO experiments accounted for 100M timesteps, though most seeds converged near 40M. The larger budget was not intended to confirm the sample complexity of the task but rather to confirm the convergence under the reward function. Nevertheless, 40M samples for a two-dimensional benchmark remain a relatively high sample complexity. In the Müller-Brown environment, this does not present a practical limitation because the potential is computationally cheap. However, on a real chemical surface, where every evaluation is expensive, that same sample count would be prohibitive. Therefore, improving it through hyperparameter tuning or more sample-efficient off-policy methods is left for future work.

Finally, the budget above concerns how many samples are needed. A separate cost is how expensive each sample is to evaluate. The existing deterministic NEB method relies on an expensive energy and force evaluation at every image until convergence, and it repeats this cost in full for every new system. RL faces the same evaluation cost during training, but it can compensate for it by learning an energy model as an auxiliary task. It is possible to train the policy network and the energy predictor model at the same time during the rollout collections, since the models might share representation layers. Once the model is accurate, the agent can use it to get the potential value, which cuts the number of expensive evaluations. Unlike NEB, which cannot reuse

its evaluations across systems, a learned model transfers, so the saving compounds as it is applied to new instances. The true potential is still needed during training, so the benefit appears only once the model generalizes.

7 Conclusions

This thesis studies reinforcement learning (RL) as a way to recover minimum energy paths (MEPs) on a potential energy surface (PES). The search is framed as a sequential decision problem, in which a reward function guides transitions from a start state to a target state at minimal cost. Three reward designs are compared on the Müller-Brown potential across several discount factors, using value iteration to find the exact solution on the discretized landscape and the nudged elastic band (NEB) path as a reference. For the continuous states setting, the qualitatively best reward is chosen and carried for training a PPO agent with 5 seeds. The results show that the two formulations currently used in the field, the negative energy of the next state and the negative energy difference between consecutive states, do not recover the true MEP. They return short trajectories due to a structure failure rather than a matter of tuning. In contrast, the proposed reward $\mathcal{R}^+$, which penalizes only uphill moves, matches the NEB path in 3/5 cases and the transition states in 4/5. In general, it succeeds when the path has a single barrier and the target lies below the start state, and breaks down once the path requires a committed descent into an intermediate minimum. Overall, the question of how to recover a MEP on a PES with RL remains open, although this thesis takes a step toward an answer.

References

[1] Naman Katyal, Chunhui Li, Martin Kunz, Simon J. Teat, Piotr Zarzycki, Gerbrand Ceder, and Michael L. Whittaker. Defect-mediated diffusion pathways in spodumene accelerate lithium transport. *ACS Materials Letters*, 7(10):3388–3393, 2025.

[2] Jiajun Lu, David Scheerer, Gilad Haran, Wenfei Li, and Wei Wang. Role of repeated conformational transitions in substrate binding of adenylate kinase. *The Journal of Physical Chemistry B*, 126(41):8188–8201, 2022. PMID: 36222098.

[3] Qing Zhao, John Mark P. Martirez, and Emily A. Carter. Revisiting understanding of electrochemical co2 reduction on cu(111): Competing proton-coupled electron transfer reaction mechanisms revealed by embedded correlated wavefunction theory. *Journal of the American Chemical Society*, 143(16):6152–6164, 2021. PMID: 33851840.

[4] Hannes Jónsson, Greg Mills, and Karsten W. Jacobsen. *Nudged elastic band method for finding minimum energy paths of transitions*, pages 385–404. World Scientific, 1998.

[5] Graeme Henkelman, Blas P. Uberuaga, and Hannes Jónsson. A climbing image nudged elastic band method for finding saddle points and minimum energy paths. *The Journal of Chemical Physics*, 113(22):9901–9904, Dec 2000.

[6] Weinan E, Weiqing Ren, and Eric Vanden-Eijnden. String method for the study of rare events. *Physical Review B*, 66(5), August 2002.

[7] {R. A.} Olsen, {G. J.} Kroes, G. Henkelman, A. Arnaldsson, and H. Jónsson. Comparison of methods for finding saddle points without knowledge of the final states. *Journal of Chemical Physics*, 121(20):9776–9792, November 2004.

[8] Olli-Pekka Koistinen, Freyja B. Dagbjartsdóttir, Vilhjálmur Ásgeirsson, Aki Vehtari, and Hannes Jónsson. Nudged elastic band calculations accelerated with gaussian process regression. *The Journal of Chemical Physics*, 147(15), 2017.

[9] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[10] Adittya Pal. Estimating reaction barriers with deep reinforcement learning. *Data Sci.*, 7:73–92, 2024.

[11] Klaus Müller and Leo D. Brown. Location of saddle points and minimum energy paths by a constrained simplex optimization procedure. *Theoretica chimica acta*, 53:75–93, 1979.

[12] Alexis W. Mills, Joshua J. Goings, David Beck, Chao Yang, and Xiaosong Li. Exploring potential energy surfaces using reinforcement machine learning. *Journal of Chemical Information and Modeling*, 62(13):3169–3179, 2022.

[13] Richard Bellman. *Dynamic Programming*. Dover Publications, 1957.

[14] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[15] Ken ichi Fukui. The path of chemical reactions - the irc approach. *Accounts of Chemical Research*, 14:363–368, 1981.

[16] S. V. Poluyan and N. M. Ershov. The simplified string method in the calculation of the minimum energy path. *Comput. Math. Model.*, 34(3):227–234, July 2023.

[17] Baron Peters, Andreas Heyden, Alexis T. Bell, and Arup K. Chakraborty. A growing string method for determining transition states: comparison to the nudged elastic band and string methods. *The Journal of chemical physics*, 120 17:7877–86, 2004.

[18] Robert G. Parr and Yang Weitao. *Density-Functional Theory of Atoms and Molecules*. Oxford University Press, USA, 1994.

[19] Louk Rademaker. A practical introduction to density functional theory, 2020.

[20] A. K. Rappé, C. J. Casewit, K. S. Colwell, W. A. Goddard III, and W. M. Skiff. Uff, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *Journal of the American Chemical Society*, 114(25):10024–10035, 1992.

[21] Mathias Schreiner, Arghya Bhowmik, Tejs Vegge, Peter Bjørn Jørgensen, and Ole Winther. Neuralneb – neural networks can find reaction paths fast, 2022.

[22] Chenru Duan, Guan-Horng Liu, Yuanqi Du, Tianrong Chen, Qiyuan Zhao, Haojun Jia, Carla P. Gomes, Evangelos A. Theodorou, and Heather J. Kulik. React-ot: Optimal transport for generating transition state in chemical reactions, 2024.

[23] Lagnajit Pattanaik, John B. Ingraham, Colin A. Grambow, and William H. Green. Generating transition states of isomerization reactions with deep learning. *Phys. Chem. Chem. Phys.*, 22:23618–23626, 2020.

[24] Riley Jackson, Wenyuan Zhang, and Jason Pearson. Tsnet: predicting transition state structures with tensor field networks and transfer learning. *Chem. Sci.*, 12:10022–10040, 2021.

[25] Sunghwan Choi. Prediction of transition state structures of gas-phase chemical reactions via machine learning. *Nature Communications*, 14(1):1168, 2023.

[26] Rhyan Barrett and Julia Westermayr. Reinforcement Learning for Traversing Chemical Structure Space: Optimizing Transition States and Minimum Energy Paths of Molecules. *The Journal of Physical Chemistry Letters*, 15(1):349–356, 2024.

[27] Takechika Kikkawa, Takashi Kawakami, Shusuke Yamanaka, and Mitsutaka Okumura. Transition-state search algorithm based on deep reinforcement learning. *Chemistry Letters*, 54(5):upaf098, May 2025.

[28] Semeon A. Bogaty and Alexey A. Tuzhilin. Fundamentals of theory of continuous gromov–hausdorff distance, 2025.

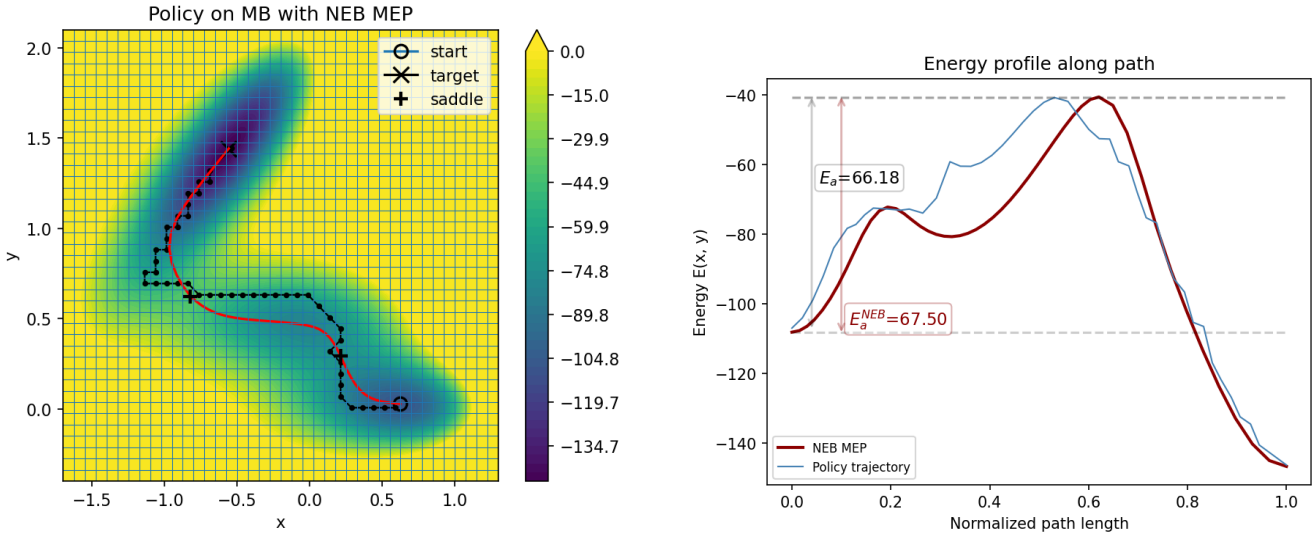
[29] Emily Fox, Amir Nayyeri, Jonathan James Perry, and Benjamin Raichel. Fréchet edit distance, 2024.

[30] Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, and Jeff Braga. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *CoRR*, abs/2111.08819, 2021.

[31] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018.

[32] David J. Wales and Tetyana V. Bogdan. Potential energy and free energy landscapes. *The Journal of Physical Chemistry B*, 110(42):20765–20776, 2006.

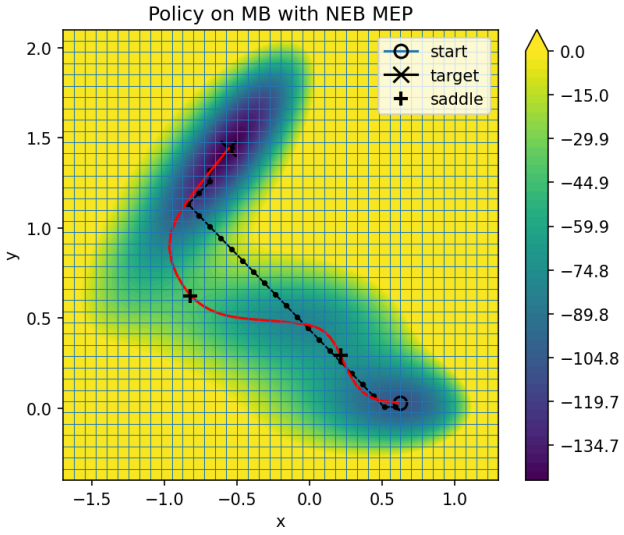
A Appendix



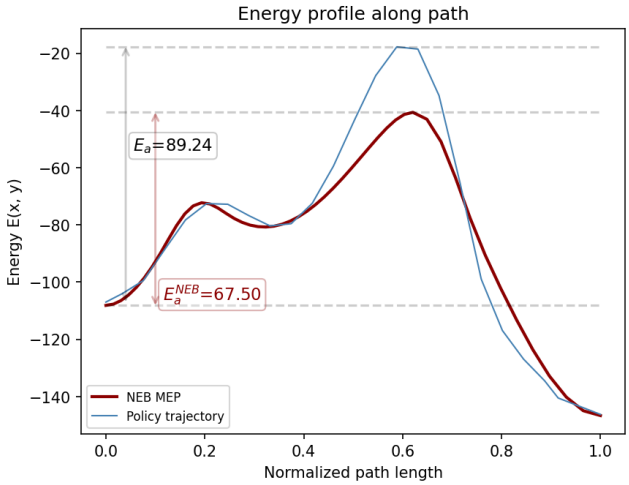
(a) Policy trajectory on the Müller–Brown surface

(b) Energy profile along the trajectory

Figure 9: Optimal policy obtained from VI with the custom reward $\mathcal{R}^+$ at $\gamma = 1$ on a finer 40×40 discretization of the Müller-Brown surface. Panel (a) shows the trajectory from the start state A to the target B, with the NEB MEP overlaid in red. Panel (b) shows the energy along the trajectory against normalized path length, compared to the NEB profile. The policy crosses at an activation energy of $E_a = 66.18$, within 1.32 of the NEB reference $E_a^{\mathrm{NEB}} = 67.50$.

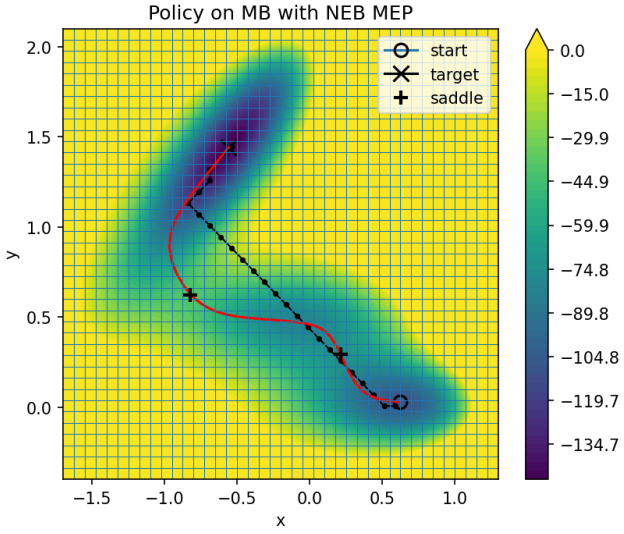


(a) Policy trajectory on the Müller-Brown surface

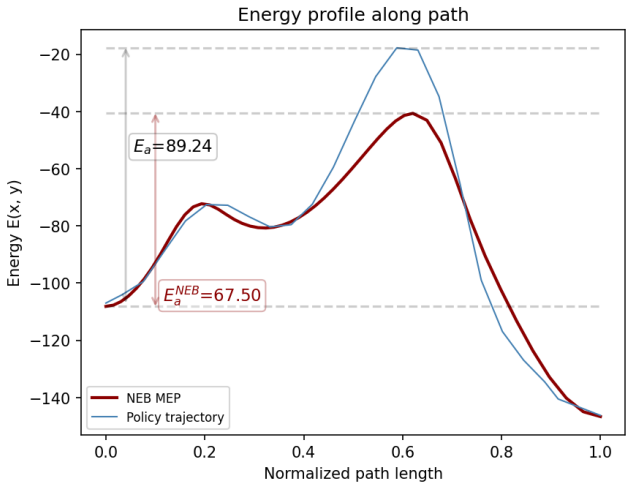


(b) Energy profile along the trajectory

Figure 10: Optimal policy obtained from VI with the literature reward $\mathcal{R}^{\Delta}$ at $\gamma = 0.99$ on a finer 40×40 discretization of the Müller-Brown surface. Panel (a) shows the trajectory from the start state A to the target B, with the NEB MEP overlaid in red. Panel (b) shows the energy along the trajectory against normalized path length, compared to the NEB profile. The policy crosses at an activation energy of $E_a = 89.24$, within 21.74 of the NEB reference $E_a^{\text{NEB}} = 67.50$.



(a) Policy trajectory on the Müller-Brown surface



(b) Energy profile along the trajectory

Figure 11: Optimal policy obtained from VI with the literature reward $\mathcal{R}^{next}$ at $\gamma = 0.99$ on a finer 40×40 discretization of the Müller-Brown surface. Panel (a) shows the trajectory from the start state A to the target B, with the NEB MEP overlaid in red. Panel (b) shows the energy along the trajectory against normalized path length, compared to the NEB profile. The policy crosses at an activation energy of $E_a = 89.24$, within 21.74 of the NEB reference $E_a^{\text{NEB}} = 67.50$.