MASTERING ATARI 2600 GAMES WITH DISCOVERED OPTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Temporal abstractions, often instantiated as options, have long been regarded as a mechanism for accelerating credit assignment, facilitating exploration, and enabling generalisation in reinforcement learning (RL). However, developing general option discovery methods that are effective in large-scale, high-dimensional domains remains a fundamental challenge. Existing option discovery methods are either confined to relatively simple domains, depend on handcrafted or quasi-symbolic representations, or offer little improvement over learning without options. We present Wayfarer, a general, domain-agnostic, online deep RL agent that discovers options through Laplacian representation learning from high-dimensional observations and leverages them for control. We show that the resulting options simultaneously improve exploration, accelerate credit assignment, and generalise effectively to unseen settings, enabling substantially faster learning of complex policies. Wayfarer achieves state-of-the-art performance among single-stream agents on the most challenging Atari 2600 games, with the largest gains in games that require long-horizon exploration and strategic behaviour, such as Montezuma's Revenge and Private Eye. Our code will be shared upon publication.

1 INTRODUCTION

We must perceive in order to move, but we must also move in order to perceive.
— J. J. Gibson [1979, p. 213]

Intelligent behaviour spans many time scales at once. Imagine travelling to a conference: you book the trip, navigate an unfamiliar city, and, far down the hierarchy, coordinate innumerable muscle twitches. Reinforcement learning (RL) commonly formalises such decision-making in discrete time steps: at each step, the agent observes its environment, takes an action, and receives a reward. Maximising this reward signal defines the agent's goal. The time step is the fundamental unit of time in this formalism, yet many intelligent behaviours are built from what the formalism itself does not immediately supply: temporally extended courses of action, each chosen as a single decision, at multiple levels of temporal abstraction.

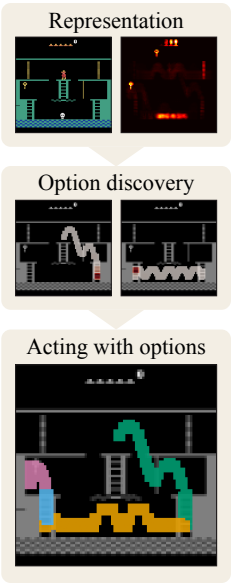


Figure 1: *Wayfarer* learns a representation, derives options from it, and learns a policy over actions and options.

Options [2, 3] are central to such abstraction in RL. An option is a skill, formalised as a policy paired with a termination condition, that the agent can use for planning, for credit assignment, or as an abstract action to follow—that is, acting according to the option's policy until it terminates. An option abstracts away low-level actions and instead represents a temporally extended behaviour—crossing the room, descending the ladder—so decisions become fewer and the intervals between them longer. Therein lies the long-recognised promise of options [4, pp. 5–6]: they facilitate exploration, because directed behaviour replaces step-by-step dithering [5]; they accelerate temporal credit assignment, because outcomes are attributed to a few long decisions rather than many short ones [2]; and they enable generalisation, because a skill learned once can transfer to situations never seen before [6].

In practice, however, options are often given, not discovered. Deep RL agents now master real-time strategy games [7], navigate stratospheric balloons [8], and outplay elite athletes [9], but where

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

such systems act at multiple time scales, the abstraction is engineered by their designers rather than learned from experience. Developing option-discovery methods that are effective at scale, in high-dimensional domains, remains a fundamental challenge.

Many existing discovery methods ground their options in the task reward [10, 11] or in features given by the designer [12, 13]. But where non-zero rewards are sparse, methods grounded in reward have no signal to build on. Handcrafted subgoals, providing their own signal, get around this. Yet to give the agent its subgoals is to assume what should be discovered, and such a method is only as general as the subgoals it is given. Finding truly general methods remains a major open question.

We tackle this question head-on by grounding option discovery in the structure of the environment, as captured by the *Laplacian representation* [14]. The Laplacian representation maps observations to a set of features that captures how slowly they vary along the agent's experience. Observations close in time map to nearby points in the representation space, and the smoothest features trace the environment's large-scale, slowest-evolving structure. Laplacian-derived options [15–25] thus need neither task rewards nor designer-given subgoals: they are grounded in the agent's observations alone.

The Laplacian representation has been used to facilitate exploration [15–20], credit assignment [21, 22], skill transfer [23, 24], and planning [25]; but each in isolation. Moreover, the representation is typically learned offline, from a pre-collected dataset, which assumes that we already have the very coverage that only good exploration could have produced. The exception that learns it online uses its options for exploration alone [26]. Indeed, to our knowledge, no option-discovery method—whether grounded in reward, in given subgoals, or in the structure of the environment itself—has been shown to be able to effectively tackle exploration, credit assignment, and generalisation at once when processing high-dimensional observations, let alone doing so online, from a single stream of experience.

We present *Wayfarer*, a general, domain-agnostic, model-free agent that closes this gap by combining modern deep RL with Laplacian representation learning. Wayfarer couples three networks trained simultaneously, online and off-policy, from a single stream of experience (Figure 1). **(1)** A representation network approximates the Laplacian representation of an abstract, *contingent-aware* feature space that represents what is under the agent's control [27, 28]. **(2)** A skill network learns a repertoire of temporally extended behaviours. **(3)** A main network learns to maximise reward over the action set augmented with these options. The networks form a **virtuous loop**: the representation defines the options' intrinsic rewards; the options augmenting the agent's action set shape the experience it collects; and this experience is used when training the networks, so the representation and the options improve one another as the agent expands the frontier of what it knows.

We show that Wayfarer delivers the promise of options: they improve exploration, accelerate credit assignment, and generalise to previously unseen settings, enabling substantially faster learning of complex policies. Three phenomena underlie these gains. **First**, the smoothest features of the representation come to identify regions in the agent's environment that are weakly connected, and learning options from this information enables the agent to navigate between them. **Second**, these options allow the agent to quickly learn a rough policy sketch that, over time, is distilled and refined into a policy over low-level primitive actions. **Third**, we show options discovered in a specific part of the environment are readily available and effective in other unseen parts of the environment.

Among agents trained on a single stream of experience, Wayfarer achieves the state of the art on the *challenging set*: ten Atari 2600 games that demand hard exploration or long-horizon credit assignment, and on which progress has historically been slow [29]. It does so with a single hyperparameter configuration across games. The largest gains appear where long-horizon exploration and strategic behaviour are required, as in *Montezuma's Revenge* and *Private Eye*.

2 Reinforcement learning with Laplacian-based options

We consider an agent learning to achieve goals by interacting with its environment in a sequence of episodes. The agent starts from an initial state, $s_0 \in \mathcal{S}$, receives an observation from the environment, $x \in \mathcal{X}$, and takes an action from the set of available actions $a \in \mathcal{A}$. The environment responds by transitioning to an unobservable state, delivering to the agent a reward signal, $r \in \mathcal{R} \subset \mathbb{R}$, and a new observation $x' \in \mathcal{X}$. This interaction continues until the episode terminates, after which the agent starts again from an initial state. The agent's goal is defined by the reward, and it learns a policy to maximise its return—the expected discounted sum of the rewards over time. An

action-value function predicts this return for an observation–action pair, which allows the agent to estimate how rewarding particular observation–action combinations are over the long run.

In the high-dimensional setting, observations can contain substantial variation unrelated to the agent's actions: the agent can perceive information through its sensory stream that it does not control. Efroni et al. [30] formalise this distinction: observations are emitted by latent states, each factoring into two components: an *endogenous* component, whose transitions depend on the agent's actions, and an *exogenous* ("noise") component, which evolves independently. We assume that such a dynamics factor exists, which motivates learning an *agent-centric* representation [31]: a mapping from the agent's observations to their endogenous components—a *contingent* feature space.

Options extend the agent's decision making across time [2, 3]. Each option $o \in \mathcal{O}$ has a policy over primitive actions and a termination function. Under *call-and-return* execution, selecting an option delegates control to that option's policy until termination; after termination, control returns to a higher-level policy over the options. A primitive action is an option of length one, so $\mathcal{A} \subset \mathcal{O}$.

The transition dynamics of an environment give the state space a shape. Under a fixed policy, they induce a weighted graph: states are nodes, and the edge weight $w_{s,s'} \geq 0$ between states s and s' reflects how directly the dynamics connect them. The adjacency matrix $\mathbf{W} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$, with $W_{s,s'} = w_{s,s'}$, contains these weights; the degree matrix $\mathbf{D}$ is the diagonal matrix of the row sums of $\mathbf{W}$; and the *graph Laplacian* for symmetric weight matrices is given by $\mathbf{L} = \mathbf{D} - \mathbf{W}$.[1] In the RL setting, the Laplacian can be defined as any matrix $\mathbf{L} = \mathbf{I} - f(\mathbf{P}_\pi)$, where $\mathbf{P}_\pi$ defines the probabilities of transitioning between states while following policy π, and where $f : \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|} \to \mathrm{Sym}_{|\mathcal{S}|}(\mathbb{R})$ maps $\mathbf{P}_\pi$ to a symmetric matrix [33]. The eigenvectors of $\mathbf{L}$ define the *Laplacian representation*, capturing the principal directions of information diffusion on the graph. The eigenvectors corresponding to the smallest eigenvalues trace the graph's large-scale temporal structure, varying slowly across strongly connected states [14]. The first eigenvector is constant; subsequent ones capture this structure ordered by smoothness (Figure 8, Appendix A). With high-dimensional observations, the agent must approximate the Laplacian representation from experience. One way to do this is to approximate it from observations using stochastic optimisation [19, 32–34]. We typically parameterise this function using a deep neural network; we refer to the outputs of such a network as *eigenfunction* estimates. We discuss specific objective functions in Section 3.

Differences in the representation across observations can be used for *intrinsic rewards* [35–39], providing the agent with a learning signal independent of task reward. Options trained to follow the gradient directions of the representation can discover routes between weakly connected regions [15, 16], empowering the agent to reach states unlikely to be encountered by dithering with primitive actions.

Deep action-value learning provides a way to learn both these option policies and when to use them. The Deep Q-Networks (DQN) algorithm [40, 41] estimates action values from pixels, learning off-policy from experience replay [42]. Rainbow [43] extends DQN with double Q-learning [44], prioritised experience replay [45], a dueling Q-network [46], multi-step learning [47], distributional RL [48], and a noisy network [49]. Wayfarer uses these components with Implicit Quantile Networks (IQN) [50]. IQN agents approximate distributional returns via learned quantile functions; such agents form an *implicit* representation of the return distribution.

3 WAYFARER

Wayfarer couples representation learning, option discovery, and task learning in a single online loop. The agent begins by acting randomly and experiencing how the environment responds. Henceforth, it instantiates the *representation-driven option discovery* (ROD) cycle [51, 52]: the experience shapes a representation; the representation defines intrinsic rewards for option discovery; and the options shape what the agent experiences in the future. Early experience covers only a small part of a large state space, but even a rough representation supports options that reach states unlikely to be encountered by dithering with primitive actions. We visualise the difference in Figure 4 (Section 4).

The algorithm consists of three neural networks (Figure 9, Appendix B). The behaviour policy is determined by the **main network**, which learns from task rewards to estimate action values over both primitive actions and options. When it selects an option, that option's policy controls the prim-

[1] We present the discrete Laplacian for ease of exposition, but use its abstract generalisation [19, 32, 33].

itive actions until termination, with a fixed termination probability at each step. The **skill network** learns a pair of options for each non-constant eigenfunction of the representation: one rewarded for increasing the value of the eigenfunction, the other for decreasing it. The **representation network** learns these eigenfunctions, approximating the Laplacian representation of the agent-centric state—capturing slowly varying structure in the parts of the agent's observations that are under its control.

All networks learn off-policy from a shared experience replay buffer, $\mathcal{D}$, and they all update within the same update step. Our learning algorithm addresses three demands of this training loop: **(1)** extracting useful structure from high-dimensional observations; **(2)** transforming that structure into coherent, directed behaviours; and **(3)** learning when these behaviours help maximise task reward.

While interacting with the environment, the agent minimises four loss functions. The main network learns action and option values to maximise task rewards (Section 3.1), while the skill network (Section 3.2) learns action values from intrinsic rewards induced by the representation learned by the representation network (Section 3.3). Both the main and skill networks use a form of *Q-learning* [53], combining rewards from replayed experience with bootstrapped estimates of future return.

For option discovery, we want the representation to capture temporal structure in aspects of the environment that the agent can influence. To achieve this, the representation network regularises its encoder using *multi-step inverse-dynamics prediction* [30, 54], which discovers a feature space of information useful for inferring the agent's actions. Using this encoder, a separate head in the network approximates the Laplacian representation through a variant of the *spectral graph-drawing* objective [55], originally formulated as a constrained optimisation problem that minimises variation across connected states subject to orthonormality constraints. This objective can be relaxed; the smoothness and constraint terms are estimated from experience, allowing the representation to be learned via stochastic optimisation [19, 32–34]. The two objectives train jointly, encouraging eigenfunctions that capture temporal structure from control-relevant information.[2]

3.1 Acting over the option set

To the agent, a primitive action is just another option in its option set, $a \in \mathcal{A} \subset \mathcal{O}$, so we denote both primitive actions and options by $o \in \mathcal{O}$. The main network estimates values over $\mathcal{O}$, combining Rainbow's components with IQN. Noisy layers in the dueling value and advantage streams contribute to exploration alongside the temporally extended behaviours induced by the options.

Let $Z(x, o)$ denote the return random variable associated with observation x and option o, and let $F_{Z(x,o)}$ denote its cumulative distribution function. We write $Z(x, o, \tau) := F^{-1}_{Z(x,o)}(\tau)$ for its quantile at level $\tau \in [0, 1]$. IQN approximates this quantile function with a *universal value function approximator* [56] that maps from x, o, and $\tau \sim U[0, 1]$ to samples from the implicitly defined return distribution [50, p. 4]. A distortion function $g : [0, 1] \to [0, 1]$ transforms the sampled quantile levels, resulting in the *distorted expectation* $Q_g(x, o) := \mathbb{E}_{\tau \sim U[0,1]}\big[Z\left(x, o, g(\tau)\right)\big]$. Different choices of g give rise to a spectrum of *risk-sensitive* policies. Identity corresponds to risk-neutrality.

When no option is active, the agent selects the action or option with the highest estimate of the expected return, with noisy layers injecting exploration noise into the agent's estimates. The agent's behaviour policy is risk-neutral. To estimate the expectation, the agent draws N_{sel} independent quantile levels from $U[0, 1]$:

$$\pi(x) := \arg\max_{o \in \mathcal{O}} \frac{1}{N_{\text{sel}}} \sum_{i=1}^{N_{\text{sel}}} h^{-1}\big(Z(x, o, \tau_i)\big).$$

The value estimates are learned in a transformed space, using $h(z) := \operatorname{sign}(z)\big(\sqrt{|z| + 1} - 1\big) + \varepsilon z$ for a small constant ε. Applying h^{-1} before averaging returns them to the original scale. This transformation is used as an alternative to reward clipping, reducing the scale of the optimisation target while retaining information about reward magnitude discarded by sign clipping [29, 57–59].

Selecting option o delegates control to that option's policy, π_o, until it terminates. The probability of termination is P_{term} in every non-terminal state, giving an expected duration of $1/P_{\text{term}}$ steps.

[2] The main network has its own encoder and therefore the agent can still use information beyond its control—such as a countdown timer or passing traffic—in its higher-level decision making.

Learning from task rewards. The main network learns from prioritised experience replay using the truncated n-step return $G_t^n = \sum_{m=0}^{n-1} \gamma^m r_{t+m+1}$. The online network selects the bootstrap action, and a target network [41] evaluates it. With the online and target Z-functions parameterised by θ and θ^-, respectively, the target is

$$y_j := h\Big(G_t + \gamma^n \, h^{-1}\big(Z_{\theta^-}(x_{t+n}, o', \tau'_j)\big)\Big)\,, \quad o' := \arg\max_{o \in \mathcal{O}} \frac{1}{N_{\text{sel}}} \sum_{i=1}^{N_{\text{sel}}} h^{-1}\big(Z_\theta(x_{t+n}, o, \tau_i)\big)\,.$$

The quantile-Huber loss [60] compares these targets with the online quantile estimates, and replay importance weights adjust each sample's contribution to correct for sampling bias (Appendix D).

In online ROD, options are derived from a representation that changes continually as the agent interacts with the environment. Their policies therefore change too (the development visualised in Figure 4, Section 4), posing one of the biggest challenges in learning values over the discovered options. In particular, a previously rewarding behaviour may not remain available through the same option. Our aim is, therefore, to let expected task return guide which options to initiate and when, while distilling the useful behaviours into a policy over primitive actions as the options evolve.

To do such distillation, the agent learns as follows. When the main network initiates an option, the replay sample stores that option. At every other step, including subsequent steps during that option's execution, it stores the primitive action taken. For simplicity, the main network updates both option and action values using the fixed n-step target above, bootstrapping regardless of whether the option has terminated. Thus, task rewards train both option initiation and primitive action selection along the trajectories the options generate. Empirically, the agent appears to favour options at the frontier of its experience and primitive actions as those regions become familiar (Figure 11, Appendix C). Viewing $\mathcal{D}$ as a non-parametric model and replay as planning [61], we can interpret this shift as the agent distilling, via planning, experience generated by options into a policy over primitive actions.

3.2 From the representation to options

While the main network learns about the expected return of such options, that is, the situations that are best for using options, the skill network learns the options' policies. To do this, the network uses intrinsic rewards induced by the learned representation as goals that options learn to maximise. In particular, these intrinsic reward functions are derived from the Laplacian representation, which the agent approximates using d eigenfunctions. Each eigenfunction, u_i, after discarding the first, $i \in \{2, \cdots, d\}$, gives rise to two intrinsic reward functions:

$$r^{i+}(x_t, x_{t+n}) := u_i(x_{t+n}) - u_i(x_t)\,, \qquad r^{i-}(x_t, x_{t+n}) := -\big(u_i(x_{t+n}) - u_i(x_t)\big)\,.$$

The multi-step horizon n is shared with the main network. The one-step intrinsic rewards telescope, meaning the intermediate terms cancel, so each reward is an undiscounted sum of one-step eigenfunction changes over this horizon. The rewards are computed with the current representation, so they change as the agent learns more about the environment's structure.

The skill network learns $K = 2(d-1)$ options, off-policy, from the agent's experience replay buffer. A shared convolutional encoder thus feeds K IQN heads, each estimating primitive-action values for one intrinsic reward, and learning from uniformly sampled experience replay using n-step updates. Each option's policy acts greedily over $\mathcal{A}$ with respect to the mean of its estimated value quantiles. All K heads update at every update step, and the shared encoder descends on their averaged gradient.

3.3 From observations to the representation

The Laplacian captures changes in slowly evolving aspects of the agent's experience [62]. These changes often correspond to visiting states that are hard to reach—rare occurrences in the agent's data stream—and thus help guide exploration. However, this can create a trap: a slowly varying eigenfunction need not capture variation contingent on the agent's actions. A timer can count down or traffic can move slowly across the agent's field of view regardless of what the agent does. An option may then receive reward for these changes without influencing them. Wayfarer therefore learns an abstract agent state and a Laplacian representation of the graph induced by that state.

To do this, Wayfarer couples Laplacian representation learning with a prediction problem that requires the agent to learn about the effects of its actions on its future observations. The representation

network consists of a convolutional encoder parameterised by ψ shared by a Laplacian head parameterised by ϕ and an inverse-dynamics prediction head by χ. The two heads output eigenfunction estimates, $\mathbf{u} \in \mathbb{R}^d$, and probabilities over actions, $\hat{\mathbf{a}} \in [0,1]^{|\mathcal{A}|}$, respectively. We write u_i to denote the i-th output of the Laplacian head and $\hat{a}_i$ to denote the i-th output of the inverse-dynamics head.

Agent-centric regularisation. The agent predicts the first action of a multi-step transition from the concatenated ($\oplus$) state-feature vectors of two observations separated by Δ steps [30]. Specifically, the agent replays transitions $(x_t, a_t, x_{t+\Delta})$ from $\mathcal{D}$, with $P(\Delta) \propto \gamma_{\text{rep}}^{\Delta-1}$ truncated at the episode boundary, to minimise the cross-entropy loss:

$$\mathcal{L}_{\text{inv}}(\psi, \chi) := \mathbb{E}_{(x_t, a_t, x_{t+\Delta}) \sim P_{\text{rep}}} \big[-\log q_\chi(a_t \mid x_t, x_{t+\Delta}) \big] ,$$
$$q_\chi(\cdot \mid x_t, x_{t+\Delta}) := \text{softmax}\Big(f_\chi\big(f_\psi(x_t) \oplus f_\psi(x_{t+\Delta}) \big) \Big) ,$$

where P_{rep} is the joint distribution induced by sampling (x_t, a_t) uniformly from $\mathcal{D}$ and $\Delta \sim P(\Delta)$.

Laplacian representation learning. The Laplacian objective seeks diverse and smooth eigenfunctions. Wayfarer uses the *Augmented Lagrangian Laplacian Objective* (ALLO) [33]:

$$\max_{\boldsymbol{\beta}} \min_{\phi} \sum_{i=1}^{d} \langle u_i, L u_i \rangle + \sum_{j=1}^{d} \sum_{k=1}^{j} \beta_{jk} \big(\langle u_j, [\![u_k]\!] \rangle - \delta_{jk} \big) + b \sum_{j=1}^{d} \sum_{k=1}^{j} \big(\langle u_j, [\![u_k]\!] \rangle - \delta_{jk} \big)^2 .$$

The first term is estimated from observation pairs $(x_t, x_{t+\Delta})$ replayed from $\mathcal{D}$, with $\Delta \sim P(\Delta)$. This term penalises changes across observations over time. The remaining terms enforce orthonormality through Lagrange multipliers β_{jk} and a quadratic penalty of weight $b > 0$. The Kronecker delta δ_{jk} specifies the target inner product: one when $j = k$ and zero otherwise. Both constraint terms are asymmetric: the inner sum stops at j, and the stop-gradient $[\![\cdot]\!]$ treats u_k as constant under differentiation. This asymmetry makes the Laplacian eigenfunctions the only stable equilibrium point, avoiding convergence to arbitrary rotations of them [33].

Wayfarer holds b fixed and applies a symmetric log transformation to the representation outputs. Both choices address the changing data distribution when learning from online experience: novel observations can produce large outputs and constraint errors, which a growing penalty would amplify. The output transformation compresses extreme values, thereby stabilising the gradients of the loss.

Writing what is inside ALLO's max-min objective as $\mathcal{L}_{\text{lap}}$, the joint representation objective is

$$\max_{\boldsymbol{\beta}} \min_{\psi, \phi, \chi} \; \mathcal{L}_{\text{lap}}(\psi, \phi; \boldsymbol{\beta}, b) \; + \; w_{\text{aux}} \, \mathcal{L}_{\text{inv}}(\psi, \chi) ,$$

where w_{aux} weights the auxiliary objective. Only the shared encoder ψ receives both gradients. The resulting representation is encouraged to capture temporal structure while retaining information useful for inverse dynamics. Our experiments quantify the effect of this regularisation on the agent's performance, and Figure 16 (Appendix G) visualises the difference side-by-side.

Closing the loop. All three networks update in the same cycle. Each has its own encoder: experience and intrinsic rewards connect the networks. The main network's choices shape experience; this experience refines the representation; and the representation is used for learning options that feed back into the main network's choices. Following this virtuous loop, Wayfarer learns from its experience, online, while learning how to shape it, from a single stream of experience.

4 Experiments

We evaluate Wayfarer on the ten Atari 2600 games of the *challenging set* [29] in the Arcade Learning Environment (ALE) [63] with sticky actions, the full action set, and no access to life information [64]. We first compare Wayfarer's performance with state-of-the-art deep RL agents, Rainbow [43], IQN [29], and DreamerV3 [65], then zoom in to examine what the discovered options contribute.

4.1 Experimental protocol

We train Wayfarer with a single hyperparameter configuration across games, using five runs per game for 200M frames in the main experiments and three for 100M frames in the ablations. In aggregating results, each run contributes its mean episodic return over the final 10% of training, without

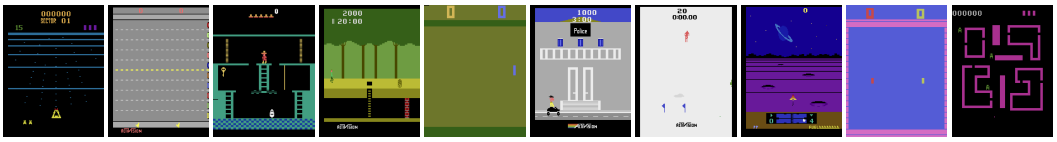


Figure 2: The Atari 2600 *challenging set*. From left to right: *Beam Rider*, *Freeway*, *Montezuma's Revenge*, *Pitfall!*, *Pong*, *Private Eye*, *Skiing*, *Solaris*, *Surround*, and *Venture*. The set was assembled by Badia et al. [29] to highlight games where progress has historically been much slower than average due to challenging exploration and credit assignment problems. Designed for entertainment, these games offer test beds of intelligent behaviour free of the experimenter's design bias [63, 66].

separate evaluation phases. Wayfarer is a *single-stream* agent: one learner, acting in one instance of the environment, learning online from the experience it generates. Rainbow and IQN provide established, state-of-the-art single-stream, model-free baselines; Dreamer provides an additional state-of-the-art reference beyond the model-free class that learns from imagined experience and collects data from parallel environment instances, placing Wayfarer's performance in a broader context. We report *human-normalised scores* [41] using the median, interquartile mean (IQM), mean, and optimality gap, with 95% bootstrapped confidence intervals [67]. Further experimental details on hyperparameters, baselines, and evaluation are provided in Appendices D, E, F, respectively.

4.2 Results and Ablations

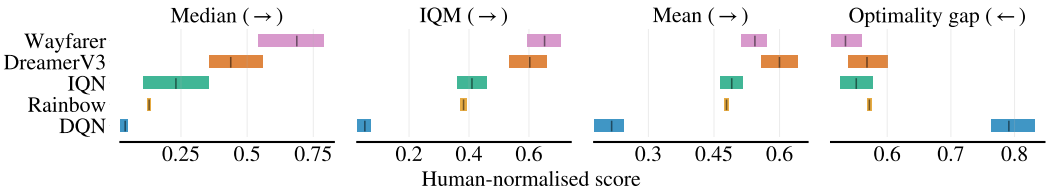


Figure 3: Aggregate metrics on the Atari 2600 challenging set, over 200M frames, with 95% CIs. Higher mean, median, and IQM and lower optimality gap are better. We report results on ten games with five runs per game, except for DreamerV3, for which we use four—the minimum available number of published runs available across games.

Figure 3 shows that **(1)** Wayfarer outperforms the model-free baselines; **(2)** it reaches the performance of Dreamer; and **(3)** all agents struggle on this benchmark, a testament to its difficulty. These results show that there is significant room for single-stream agents to improve on this set of games.

The per-game results help explain these differences. Wayfarer's performance improves with the robustness of the summary statistic (mean 0.54, IQM 0.65, median 0.69), suggesting that its performance is pulled down by outliers where it performs poorly and that it performs robustly otherwise. Of the five games where the human performs at least as well as the agents, Wayfarer outperforms its fellow agents in Montezuma's Revenge and Private Eye, Dreamer in Solaris, and Rainbow in Skiing; no agent achieves a positive return in Pitfall!. Dreamer scores exceptionally well in Beam Rider, contributing to its advantage under the mean. Wayfarer leads in Venture. We provide per-game training curves in Figure 14 and reference scores in Table 3 (Appendix F).

What are options good for? Recall the long-recognised promise of options that they improve exploration, because coherent, temporally extended behaviours replace step-by-step dithering; accelerate credit assignment, because credit can be assigned to whole behaviours; and generalise to unseen settings, allowing skills learned in one setting to be reused in others. While prior work has often focused on demonstrating these benefits in isolation, Wayfarer reaps all three at once, enabling substantially faster learning. Next, we provide anecdotal evidence to support our claims.

Options improving exploration. Most RL algorithms rely on local, single-step exploration approaches. Wayfarer mixes temporally extended exploration, designed to take the agent to the states that seem far away from the agent, with local exploration designed to provide coverage of the states around the agent's location. To demonstrate how options discovered by Wayfarer improve exploration, we disabled option termination to inspect the temporally extended behaviours they had learned. We draw examples from two games among the hardest to explore: Private Eye and Pitfall!.

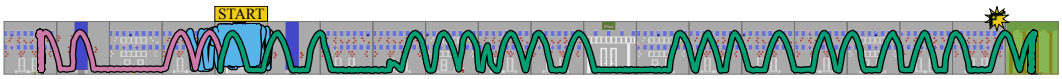


Figure 4: A single option discovered by Wayfarer, in Private Eye, at successive stages of training. The option corresponds to the smoothest eigenfunction, which captures the slowest-changing structure. The option is initialised as random (blue), quickly expands coverage (pink), and soon reaches a hard exploration-milestone (green): the option obtains a key item in the investigator's case (gun).

First, in Private Eye, a single option learns to navigate streets, parks, and secret passages while avoiding dead ends for almost two minutes of gameplay.△ Figure 4 shows this option from three earlier points in training. By 6% of the training budget, the option has reached a hard exploration-milestone: it directs the investigator all the way to the gun, a key milestone in solving the task. In comparison, obtaining the gun takes ∼600 correctly sequenced decisions using primitive actions.

Second, Figure 5 shows three options in Pitfall!. To our knowledge, no single-stream agent, including Wayfarer, has achieved a positive return in the game. Yet, by 20% of the training budget, the agent has discovered skills that master non-trivial elements of the game.△ One option crosses a deadly puddle; another leaps over holes in the ground, climbing back up each time it falls underground. The third travels underground, a direction not covered by the other two.

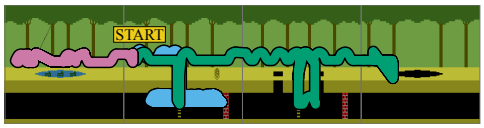


Figure 5: Three options discovered by Wayfarer in Pitfall!. One option (pink) travels left, crossing a deadly puddle in a coordinated sequence. Another (green) travels right, crossing holes and using ladders where needed. The third (blue) underground.

Options accelerating credit assignment. To evidence that options accelerate credit assignment, we compare Wayfarer's performance against a variant of it that does not learn option values, using the discovered options only as part of the agent's exploratory step. This variant is based on Deep Covering Eigenoptions (DCEO) [26]. DCEO agents use options as part of the agent's exploratory step while following an ϵ-greedy policy. When exploring, an option is chosen at random and its policy followed until termination. To match DCEO, we change Wayfarer's configuration to use an ϵ-greedy policy and the main network to only learn action values over primitive actions. The results (*Wayfarer 100M* versus *no option values* in Figure 7) clearly favour Wayfarer.

These results may be explained in part by the pattern in Figure 11 (Appendix C), which visualises empirical option-selection rates over training checkpoints and rooms from a run of Montezuma's Revenge. The heatmaps show that the agent has a strong preference for options shortly after encountering a new room: the pattern suggests that the agent is using the options to expand its frontier and then distils a fine-grained policy over primitive actions from the data generated by the options.

The bottom panel of our front-page figure (Figure 1) further exemplifies how options support credit assignment. The panel shows how options discovered in the first room of Montezuma's Revenge can be stitched together to reach the key via a handful of high-level decisions, compared to ∼70 decisions via primitive actions.△ Even with a fixed update horizon, these directed sequences of actions effectively shorten the delay of the agent's reinforcement, likely helping the agent assign credit.

Options generalising to unseen settings. To show that options generalise to unseen settings, we study how the discovered options behave in regions of Montezuma's Revenge not yet experienced by the agent. We choose a checkpoint at 0.4% of the training budget, when we can be certain that the agent has experienced only the first room, and show how two selected options from this checkpoint transfer to unseen parts of the game (Figure 6).

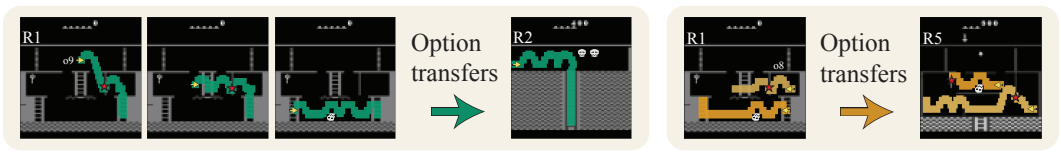


Figure 6: Options discovered in the first room of Montezuma's Revenge use ropes and avoid deadly skulls in unseen rooms without further learning. *(Left)* Option #9 moves right and descends ladders. *(Right)* Option #8 moves left, also using a rope-like pole along the way. Starting points/directions are marked with yellow arrows, ropes and poles with red stars, and deadly enemies with skull icons.

△The triangle symbol indicates that a corresponding video is available in the supplementary material.

Option #9 moves Panama Joe to the right and down the ladder, a behaviour consistent from multiple starting states. The skill remains the same in the next room. Option #8 moves Panama Joe to the left via any ropes that are encountered along the way. The skill remains the same in room five: from multiple starting states, the option travels left and is attracted to ropes and a rope-like pole. Both options also allow Panama Joe to avoid deadly enemies (skulls) in unseen rooms, a valuable skill in a game where dying is easy. A map of the game provides spatial context (Figure 10, Appendix C.1).

Ablations. Our results suggest that options discovered by Wayfarer deliver on their promise: they improve exploration, accelerate credit assignment, and transfer to unseen settings. Next, we examine how they interact with Wayfarer's design choices. Specifically, we ask four questions. **(1)** Can our good performance relative to baselines be attributed to options, or is it mostly explained by other parts of our algorithm? **(2)** How important is learning the Laplacian from the agent-centric state as opposed to from raw observations? **(3)** Does learning option values actually help or hinder performance compared with using the options only for exploration with an ϵ-greedy policy? **(4)** How does exploration through noisy value estimates compare with ϵ-greedy exploration when option values are learned in both cases? We tuned our ablated configurations; this is discussed in Appendix E.

We make the following observations from the aggregate results in Figure 7 and the per-game training curves in Figure 15 (Appendix F). **(1)** Learning options contributes substantially to Wayfarer's success: at 100M frames, a 2.51x improvement according to the median, 1.44x according to the IQM, and 1.21x according to the mean. **(2)** Learning the Laplacian representation of an abstract, contingent-aware feature space is helpful. We illustrate the difference using feature attribution in Appendix G, which makes this result rather intuitive. **(3)** Using options only for exploration performs competitively in the games that are hardest to explore but performs worse overall. **(4)** The noisy network helps the agent compared with a prescribed ϵ-greedy exploration schedule. One possible explanation is that under ϵ-greedy the agent may be forced to commit to temporally extended behaviour even when it is not needed. Note that, with ϵ-greedy exploration in both configurations (*no noisy networks* and *no option values*), learning option values plausibly improves aggregate performance slightly, which is consistent with our interpretation that options accelerate credit assignment.

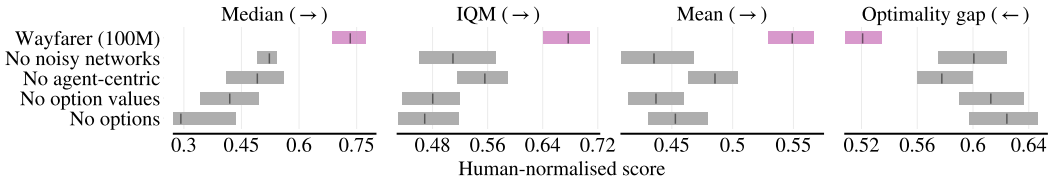


Figure 7: Aggregate metrics on the Atari 2600 challenging set, over 100M frames, with 95% CIs. Higher mean, median, and IQM and lower optimality gap are better. Ten games; three runs per game.

5 Conclusion

We have introduced Wayfarer, an online deep RL agent that discovers and learns to use options from high-dimensional observations. On the Atari 2600 challenging set, the options discovered by Wayfarer deliver on their promise: they improve exploration, accelerate credit assignment, and generalise to unseen settings. With these options, Wayfarer learns faster and achieves higher aggregate returns, surpassing the single-stream baselines IQN and Rainbow and matching the parallelised, model-based DreamerV3. These behaviours emerge from a single stream of interaction with the environment, without domain-specific knowledge or semantic supervision.

Making Wayfarer work required tackling problems such as learning option values with non-linear function approximation [68], stabilising online Laplacian representation learning [34], and fusing Laplacian representation learning with agent-centric representation learning [30, 31, 69]. Throughout the paper, we have discussed the work most closely related to ours; since the option-discovery literature is vast, we include an extended related work in Appendix H. Based on our experiments, incorporating memory beyond frame stacking, sustaining plasticity (Appendix I), and planning with discovered options [70] offer clear directions for extending this work. We conclude that Wayfarer is the beginning of something bigger: a prototype for a general agent that reasons at multiple levels of temporal abstraction and continually discovers new knowledge from experience in the environment.

AI USE STATEMENT

In this work, we used generative AI tools to polish our writing and to support most tasks involving the production of code (implementation, figures). The rest of the required disclosure tasks listed in the ICLR 2027 AI Policy for Authors are not applicable to this work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have made our best effort to ensure our work is reproducible. Appendix B illustrates our network architecture, Section 3 describes our learning algorithm, and Appendix D lists our hyperparameters. Appendix F details our score-aggregation procedure and the random and human reference scores used to compute human-normalised scores. We will publish our code upon publication of this paper.

REFERENCES

[1] James Jerome Gibson. *The ecological approach to visual perception*. Houghton Mifflin, 1979.

[2] Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1): 181–211, 1999. URL https://www.sciencedirect.com/science/Article/pii/S0004370299000521.

[3] Doina Precup. *Temporal abstraction in reinforcement learning*. PhD thesis, University of Massachusetts Amherst, 2000. URL http://www.incompleteideas.net/papers/Precup_PhD_thesis.pdf.

[4] Martin Klissarov, Akhil Bagaria, Ziyan Luo, George Konidaris, Doina Precup, and Marlos C. Machado. Discovering temporal structure: an overview of hierarchical reinforcement learning. *arXiv preprint arXiv:2506.14045*, 2025. URL https://arxiv.org/abs/2506.14045.

[5] Marlos C. Machado. *Efficient exploration in reinforcement learning through time-based representations*. PhD thesis, University of Alberta, Canada, 2019. URL https://doi.org/10.7939/r3-zaq3-vs36.

[6] George D. Konidaris and Andrew G. Barto. Building portable options: skill transfer in reinforcement learning. In *International Joint Conference on Artificial Intelligence*, 2007. URL https://www.ijcai.org/Proceedings/07/Papers/144.pdf.

[7] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell et al. Grandmaster level in StarCraft ii using multi-agent reinforcement learning. *Nature*, 575:350–354, 2019. URL https://doi.org/10.1038/s41586-019-1724-z.

[8] Marc G. Bellemare, Salvatore Candido, Pablo Samuel Castro, Jun Gong, Marlos C. Machado, Subhodeep Moitra, Sameera S. Ponda, and Ziyu Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588:77–82, 2020. URL https://doi.org/10.1038/s41586-020-2939-8.

[9] Peter Dürr, Mireille El Gheche, Guilherme Jorge Maeda, Nobuhiko Mukai, Naoya Takahashi, Stefan Heusser, Hamdi Sahloul, Yamen Saraiji et al. Outplaying elite table tennis players with an autonomous robot. *Nature*, 652:886–891, 2026. URL http://dx.doi.org/10.1038/s41586-026-10338-5.

[10] Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. FeUdal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, 2017. URL https://proceedings.mlr.press/v70/vezhnevets17a.html.

[11] Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *AAAI Conference on Artificial Intelligence*, 2017. URL http://dx.doi.org/10.1609/aaai.v31i1.10916.

[12] Satinder Singh, Andrew G Barto, and Nuttapong Chentanez. Intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems*, 2004. URL https://proceedings.neurips.cc/paper_files/paper/2004/file/4be5a36cbaca8ab9d2066debfe4e65c1-Paper.pdf.

[13] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: integrating temporal abstraction and intrinsic motivation. In *Advances in Neural Information Processing Systems*, 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/f442d33fa06832082290ad8544a8da27-Paper.pdf.

[14] Sridhar Mahadevan. Proto-value functions: developmental reinforcement learning. In *International Conference on Machine Learning*, 2005. URL https://doi.org/10.1145/1102351.1102421.

[15] Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. A Laplacian framework for option discovery in reinforcement learning. In *International Conference on Machine Learning*, 2017. URL https://proceedings.mlr.press/v70/machado17a.html.

[16] Marlos C. Machado, Clemens Rosenbaum, Xiaoxiao Guo, Miao Liu, Gerald Tesauro, and Murray Campbell. Eigenoption discovery through the deep successor representation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Bk8ZcAxR-.

[17] Yuu Jinnai, Jee Won Park, David Abel, and George Konidaris. Discovering options for exploration by minimizing cover time. In *International Conference on Machine Learning*, 2019. URL https://proceedings.mlr.press/v97/jinnai19b.html.

[18] Yuu Jinnai, Jee Won Park, Marlos C. Machado, and George Konidaris. Exploration in reinforcement learning with deep covering options. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeIyaVtwB.

[19] Kaixin Wang, Kuangqi Zhou, Qixin Zhang, Jie Shao, Bryan Hooi, and Jiashi Feng. Towards better Laplacian representation in reinforcement learning with generalized graph drawing. In *International Conference on Machine Learning*, 2021. URL https://proceedings.mlr.press/v139/wang21ae.html.

[20] Andreas Theophilou and Özgür Şimşek. Novel exploration via orthogonality. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/bc5d44c37a9ccdf9300637a1d79ad9b8-Paper-Conference.pdf.

[21] Sridhar Mahadevan and Mauro Maggioni. Proto-value functions: a Laplacian framework for learning representation and control in Markov decision processes. *Journal of Machine Learning Research*, 8(74):2169–2231, 2007. URL http://jmlr.org/papers/v8/mahadevan07a.html.

[22] Jesse Farebrother, Joshua Greaves, Rishabh Agarwal, Charline Le Lan, Ross Goroshin, Pablo Samuel Castro, and Marc G Bellemare. Proto-Value Networks: scaling representation learning with auxiliary tasks. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=oGDKSt9JrZi.

[23] Ahmed Touati, Jérémy Rapin, and Yann Ollivier. Does zero-shot reinforcement learning exist? In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=MYEap_OcQI.

[24] Siddarth Chandrasekar and Marlos C. Machado. The Laplacian keyboard: beyond the linear span. *arXiv preprint arXiv:2602.07730*, 2026. URL https://arxiv.org/abs/2602.07730.

[25] Dikshant Shehmar, Matthew Kyle Schlegel, Matthew E. Taylor, and Marlos C. Machado. Laplacian representations for decision-time planning. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=6JwkAlGLjT.

[26] Martin Klissarov and Marlos C. Machado. Deep Laplacian-based options for temporally-extended exploration. In *International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/klissarov23a.html.

[27] John S. Watson. The development and generalization of "contingency awareness" in early infancy: some hypotheses. *Merrill-Palmer Quarterly of Behavior and Development*, 12(2): 123–135, 1966. URL https://www.jstor.org/stable/23082793.

[28] Marc Bellemare, Joel Veness, and Michael Bowling. Investigating contingency awareness using Atari 2600 games. In *AAAI Conference on Artificial Intelligence*, 2012. URL https://dx.doi.org/10.1609/aaai.v26i1.8321.

[29] Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Zhaohan Daniel Guo, and Charles Blundell. Agent57: outperforming the Atari human benchmark. In *International Conference on Machine Learning*, 2020. URL https://proceedings.mlr.press/v119/badia20a.html.

[30] Yonathan Efroni, Dipendra Misra, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Provably filtering exogenous distractors using multistep inverse dynamics. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RQLLzMCefQu.

[31] Riashat Islam, Manan Tomar, Alex Lamb, Yonathan Efroni, Hongyu Zang, Aniket Rajiv Didolkar, Dipendra Misra, Xin Li et al. Principled offline RL in the presence of rich exogenous information. In *International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/islam23a.html.

[32] Yifan Wu, George Tucker, and Ofir Nachum. The Laplacian in RL: learning representations with efficient approximations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJlNpoA5YQ.

[33] Diego Gomez, Michael Bowling, and Marlos C. Machado. Proper Laplacian representation learning. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7gLfQT52Nn.

[34] Maheed H. Ahmed, Jayanth Bhargav, and Mahsa Ghasemi. Online Laplacian-based representation learning in reinforcement learning. In *International Conference on Machine Learning*, 2025. URL https://proceedings.mlr.press/v267/ahmed25a.html.

[35] Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? A typology of computational approaches. *Frontiers in Neurorobotics*, 1:1–14, 2007. URL https://doi.org/10.3389/neuro.12.006.2007.

[36] Cam Linke, Nadia M. Ady, Martha White, Thomas Degris, and Adam White. Adapting behavior via intrinsic reward: a survey and empirical study. *Journal of Artificial Intelligence Research*, 69:1287–1332, 2020. URL https://doi.org/10.1613/jair.1.12087.

[37] Cédric Colas, Tristan Karch, Olivier Sigaud, and Pierre-Yves Oudeyer. Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey. *Journal of Artificial Intelligence Research*, 74:1159–1199, 2022. URL https://doi.org/10.1613/jair.1.13554.

[38] Arthur Aubret, Laetitia Matignon, and Salima Hassas. An information-theoretic perspective on intrinsic motivation in reinforcement learning: a survey. *Entropy*, 25(2), 2023. URL https://doi.org/10.3390/e25020327.

[39] Aly Lidayan, Michael D Dennis, and Stuart Russell. BAMDP shaping: a unified theoretical framework for intrinsic motivation and reward shaping. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tijmpS9Vy2.

[40] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. URL https://arxiv.org/abs/1312.5602.

[41] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. URL http://dx.doi.org/10.1038/nature14236.

[42] Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321, 1992. URL http://dx.doi.org/10.1007/BF00992699.

[43] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot et al. Rainbow: combining improvements in deep reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2018. URL https://doi.org/10.1609/aaai.v32i1.11796.

[44] Hado Hasselt. Double Q-learning. In *Advances in Neural Information Processing Systems*, 2010. URL https://proceedings.neurips.cc/paper_files/paper/2010/file/091d584fced301b442654dd8c23b3fc9-Paper.pdf.

[45] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations*, 2016. URL https://openreview.net/forum?id=pBbWjZdoRiN.

[46] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International Conference on Machine Learning*, 2016. URL https://proceedings.mlr.press/v48/wangf16.html.

[47] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988. URL http://dx.doi.org/10.1007/BF00115009.

[48] Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, 2017. URL https://proceedings.mlr.press/v70/bellemare17a.html.

[49] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Matteo Hessel, Ian Osband, Alex Graves, Volodymyr Mnih et al. Noisy networks for exploration. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rywHCPkAW.

[50] Will Dabney, Georg Ostrovski, David Silver, and Remi Munos. Implicit Quantile Networks for distributional reinforcement learning. In *International Conference on Machine Learning*, 2018. URL https://proceedings.mlr.press/v80/dabney18a.html.

[51] Marlos C. Machado, Andre Barreto, Doina Precup, and Michael Bowling. Temporal abstraction in reinforcement learning with the successor representation. *Journal of Machine Learning Research*, 24(80):1–69, 2023. URL http://jmlr.org/papers/v24/21-1213.html.

[52] Marlos C. Machado. Representation-driven option discovery in reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2025. URL https://doi.org/10.1609/aaai.v39i27.35100.

[53] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3–4):279–292, 1992. URL http://dx.doi.org/10.1007/BF00992698.

[54] Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016. URL https://doi.org/10.48550/arXiv.1611.07507.

[55] Yehuda Koren. On spectral graph drawing. In *International Conference on Computing and Combinatorics*, 2003. URL https://dl.acm.org/doi/10.5555/1756869.1756936.

[56] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International Conference on Machine Learning*, 2015. URL https://proceedings.mlr.press/v37/schaul15.html.

[57] Tobias Pohlen, Bilal Piot, Todd Hester, Mohammad Gheshlaghi Azar, Dan Horgan, David Budden, Gabriel Barth-Maron, Hado van Hasselt et al. Observe and look further: achieving consistent performance on Atari. *arXiv preprint arXiv:1805.11593*, 2018. URL https://arxiv.org/abs/1805.11593.

[58] Steven Kapturowski, Georg Ostrovski, Will Dabney, John Quan, and Remi Munos. Recurrent experience replay in distributed reinforcement learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=r1lyTjAqYX.

[59] Adrià Puigdomènech Badia, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, Bilal Piot, Steven Kapturowski, Olivier Tieleman, Martin Arjovsky et al. Never Give Up: learning directed exploration strategies. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Sye57xStvB.

[60] Peter J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964. URL http://dx.doi.org/10.1214/aoms/1177703732.

[61] Harm Vanseijen and Rich Sutton. A deeper look at planning as learning from replay. In *International Conference on Machine Learning*, 2015. URL https://proceedings.mlr.press/v37/vanseijen15.html.

[62] Henning Sprekeler. On the relation of slow feature analysis and Laplacian eigenmaps. *Neural Computation*, 23(12):3287–3302, 2011. URL http://dx.doi.org/10.1162/NECO_a_00214.

[63] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The Arcade Learning Environment: an evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013. URL https://doi.org/10.1613/jair.3912.

[64] Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the Arcade Learning Environment: evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018. URL https://dx.doi.org/10.1613/jair.5699.

[65] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025. URL http://dx.doi.org/10.1038/s41586-025-08744-2.

[66] Kelsey Allen, Franziska Brändle, Matthew Botvinick, Judith E. Fan, Samuel J. Gershman, Alison Gopnik, Thomas L. Griffiths, Joshua K. Hartshorne et al. Using games to understand the mind. *Nature Human Behaviour*, 8(6):1035–1043, 2024. URL http://dx.doi.org/10.1038/s41562-024-01878-9.

[67] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/f514cec81cb148559cf475e7426eed5e-Paper.pdf.

[68] Harshil Kotamreddy and Marlos C. Machado. A study of value-aware eigenoptions. *arXiv preprint arXiv:2507.09127*, 2025. URL https://arxiv.org/abs/2507.09127.

[69] Alex Lamb, Riashat Islam, Yonathan Efroni, Aniket Rajiv Didolkar, Dipendra Misra, Dylan J Foster, Lekan P Molu, Rajan Chari et al. Guaranteed discovery of control-endogenous latent states with multi-step inverse models. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=TNocbXm5MZ.

[70] Richard S. Sutton, Marlos C. Machado, G. Zacharias Holland, David Szepesvari, Finbarr Timbers, Brian Tanner, and Adam White. Reward-respecting subtasks for model-based reinforcement learning. *Artificial Intelligence*, 324, 2023. URL http://dx.doi.org/10.1016/j.artint.2023.104001.

[71] Diederik P Kingma and Jimmy Ba. Adam: a method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

[72] John Quan and Georg Ostrovski. DQN Zoo: reference implementations of DQN-based agents. Software, 2020. URL http://github.com/deepmind/dqn_zoo. Version 1.2.0.

[73] Will Dabney, Georg Ostrovski, and Andre Barreto. Temporally-extended ε-greedy exploration. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=ONBPHFZ7zG4.

[74] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart et al. Mastering Atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020. URL http://dx.doi.org/10.1038/s41586-020-03051-4.

[75] Özgür Şimşek and Andrew Barto. Skill characterization based on betweenness. In *Advances in Neural Information Processing Systems*, 2008. URL https://proceedings.neurips.cc/paper_files/paper/2008/file/934815ad542a4a7c5e8a2dfa04fea9f5-Paper.pdf.

[76] Jacob Hilton. A basis-invariant version of attribution. Machine learning notes, 2024. URL https://www.jacobh.co.uk/basis_invariant_attribution.pdf. Accessed on February 16, 2026.

[77] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011. URL http://dx.doi.org/10.1137/090771806.

[78] Priyesh Vijayan, Élodie Côté-Gauthier, Mathieu Reymond, Sarath Chandar, Doina Precup, and Isabeau Prémont-Schwarz. Addressing sparse-rewards in RL with scalable hierarchical novel eigen options. In *Reinforcement Learning Beyond Rewards Workshop at the Reinforcement Learning Conference*, 2026. URL https://openreview.net/forum?id=Bn5TzXbC6s.

[79] Matteo Gallici, Mattie Fellows, Benjamin Ellis, Bartomeu Pou, Ivan Masmitja, Jakob Nicolaus Foerster, and Mario Martin. Simplifying deep temporal difference learning. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=7IzeL0kflu.

[80] Peter Dayan. Improving generalization for temporal difference learning: the successor representation. *Neural Computation*, 5(4):613–624, 1993. URL https://doi.org/10.1162/neco.1993.5.4.613.

[81] Varun Raj Kompella, Marijn Stollenga, Matthew Luciw, and Juergen Schmidhuber. Continual curiosity-driven skill acquisition from high-dimensional video inputs for humanoid robots. *Artificial Intelligence*, 247:313–335, 2017. URL http://dx.doi.org/10.1016/j.artint.2015.02.001.

[82] Seohong Park, Oleh Rybkin, and Sergey Levine. METRA: scalable unsupervised RL with metric-aware abstraction. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=c5pwL0Soay.

[83] Chongyi Zheng, Jens Tuyls, Joanne Peng, and Benjamin Eysenbach. Can a MISL fly? analysis and ingredients for mutual information skill learning. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xoIeVdFO7U.

[84] Akram Erraqabi, Marlos C. Machado., Mingde Zhao, Sainbayar Sukhbaatar, Alessandro Lazaric, Denoyer Ludovic, and Yoshua Bengio. Temporal abstractions-augmented temporally contrastive learning: An alternative to the Laplacian in RL. In *Conference on Uncertainty in Artificial Intelligence*, 2022. URL https://proceedings.mlr.press/v180/erraqabi22a.html.

[85] Ofir Nabati, Bo Dai, Shie Mannor, and Guy Tennenholtz. Spectral Bellman method: unifying representation and exploration in RL. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tHmiydOQhn.

[86] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, 2017. URL https://proceedings.mlr.press/v70/pathak17a.html.

[87] Mahsa Bastankhah, Sophie Broderick, and Benjamin Eysenbach. Learning to perceive the world through control: empowerment-based representation learning. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ZAZb1ZGnFH.

[88] Hon Tik Tse, Siddarth Chandrasekar, and Marlos C. Machado. Reward-aware proto-representations in reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=8WdkG7g6Az.

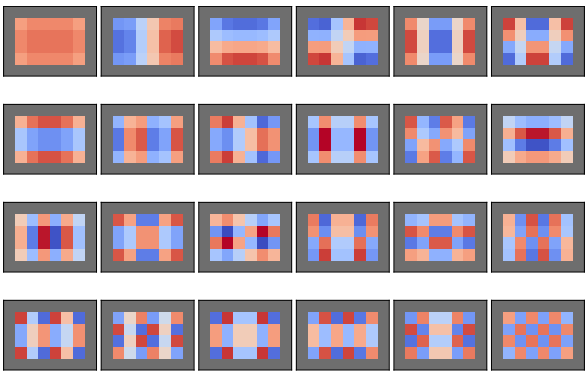
[89] Akhil Bagaria, Anita De Mello Koch, and George Konidaris. Going beyond state-reaching: Learning abstractions for intrinsically motivated option discovery. In *Reinforcement Learning in Big Worlds Workshop at RLC 2026*, 2026. URL https://openreview.net/forum?id=nhwBVHYIlG.

[90] Alper Ahmetoglu, Steven James, Cameron Allen, Sam Lobel, David Abel, and George Konidaris. Skill-driven neurosymbolic state abstractions. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=ggP2r9S9wY.

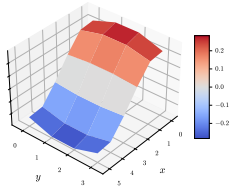
[91] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. URL https://arxiv.org/abs/1607.06450.

[92] Clare Lyle, Zeyu Zheng, Khimya Khetarpal, James Martens, Hado van Hasselt, Razvan Pascanu, and Will Dabney. Normalization and effective learning rates in reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=ZbjJE6Nq5k.

A Examples of the Laplacian representation



(a) The 24 eigenvectors in increasing order of eigenvalue. The eigenvectors corresponding to the smallest non-zero eigenvalues capture large-scale variation across the grid.

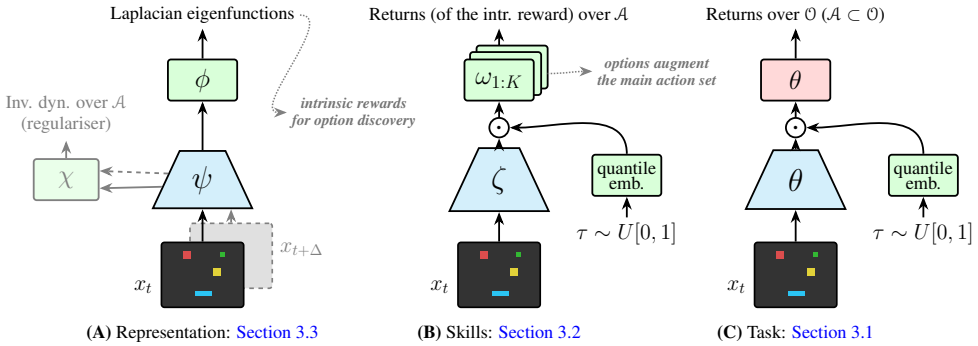


(b) The second eigenvector plotted in three dimensions to better illustrate its topology.

Figure 8: Examples of Laplacian eigenvectors from a 4×6 grid environment.

B The three Wayfarer networks

Although the three networks in Figure 9 are disjoint, they function synergistically throughout the representation-driven option discovery (ROD) cycle: the agent-centric Laplacian representation defines the intrinsic reward functions for option discovery, and the discovered options shape the agent's experiences. These experiences are used to refine the representation, and the cycle continues in a virtuous loop.



(A) Representation: Section 3.3 **(B)** Skills: Section 3.2 **(C)** Task: Section 3.1

Figure 9: The three Wayfarer networks: **(A)** The agent-centric Laplacian representation network, whose outputs are used to provide intrinsic rewards for training **(B)** the skill network, which learns the options that augment the action set of **(C)** the main network learning the extrinsic task. Blue blocks are convolutional components; green ones are linear blocks; red ones are their noisy variants.

C INVESTIGATING THE AGENT'S PREFERENCES

To inspect the agent's decision making and value preferences, we compute the empirical option-selection rates in Montezuma's Revenge from experience and network checkpoints saved during training. First, we explain the basic mechanics of the game.

C.1 MONTEZUMA'S REVENGE: GAME MECHANICS AND LEVEL ONE MAP

Figure 10 shows the first level of the game: the agent starts in the upper centre of the triangle map. To proceed in the game, the agent must collect keys to open doors while avoiding various death traps. Additionally, to accumulate more score, the agent can collect gems and slay enemies with a sword. To see in the dark, the agent can seek out a hidden torch; and so on. The agent can reach level two by making its way to the blue room in the bottom left of the triangle. To pass the level, the agent has to collect three out of the four keys and refrain from using two of them until the third-to-last room. To do this, the agent has to visit at least 15 out of the 24 rooms.

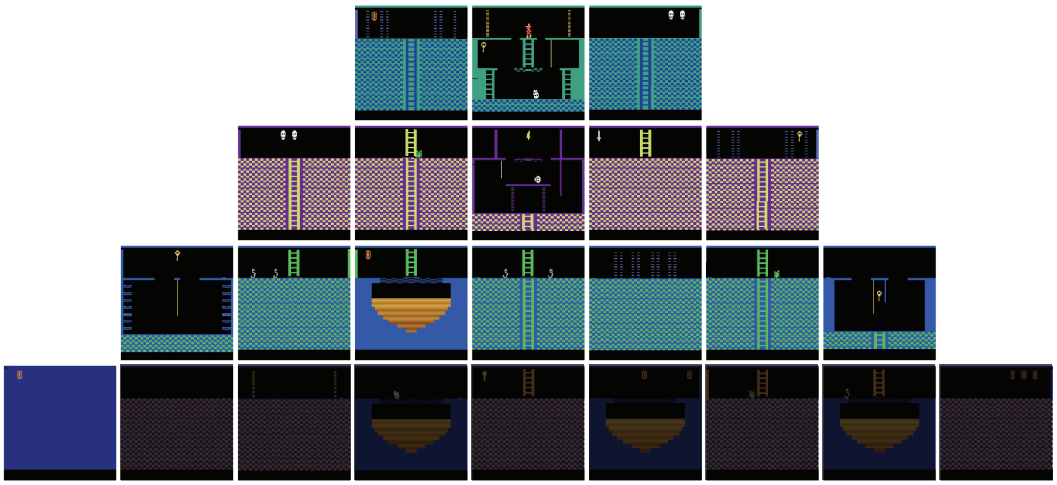


Figure 10: The first level of Montezuma's Revenge. Rooms that appear dark to the agent without a torch in its inventory are marked here with a black overlay. In game, the rooms are numbered from 0 to 23, row-major. The agent starts from room 1, and the level ends in room 15. To pass the level, at least three of the four keys have to be collected, which requires visiting at least 15 of the 24 rooms.

C.2 EMPIRICAL OPTION-SELECTION RATES

We compute a quantity that we call the empirical option-selection rate (EOSR) as follows. For each saved checkpoint, consisting of observations in the experience replay buffer and network parameters, the observations in the agent's experience replay buffer are grouped by room. Each observation is passed through the main network, and EOSR is computed for each room as the fraction of observations for which the network selects an option over a primitive action. If this quantity is higher than $K/A+K$, where K is the number of options and A the number of primitive actions, then the agent has a preference for options over primitive actions relative to uniform selection over the augmented action set. Conversely, if the quantity is lower, then the agent had a preference for primitive actions over temporally extended ones relative to the same baseline.

By dividing the observations of the experience replay buffer into rooms, and by doing so for each checkpoint, we can quantify whether the agent prefers options or primitive actions in a particular room of Montezuma's Revenge at a particular point in time during training. Figure 11 visualises the EOSR for both the main network's value estimates under noise (left), which is what defines the behaviour policy in training, and the same value estimates with noise disabled (right), thus visualising its learned preferences without NoisyNet noise.

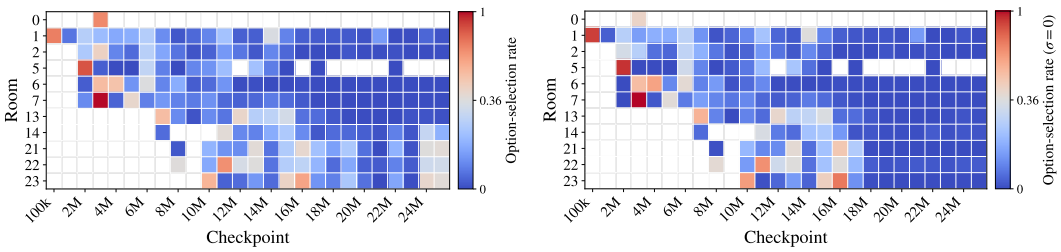


Figure 11: Empirical option-selection rates (EOSR) over the first 100M frames in Montezuma's Revenge: at each checkpoint, the observations in the agent's experience replay buffer are grouped by room; each observation is passed through the main network, and EOSR is the fraction of cases in which the network selects an option over a primitive action. Each cell quantifies whether the agent prefers options (red) or primitive actions (blue) relative to uniform selection over the 18 primitive actions and 10 options ($^{10}/_{28} \approx 0.36$), in a particular room of Montezuma's Revenge at a particular point in time during training. White cells mark rooms absent from that checkpoint's replay buffer. *(Left)* EOSR under the main network's noisy value estimates, averaged over 100 noise resamples per observation. *(Right)* EOSR with noise disabled, using the $\arg\max$ of the learned value estimates over temporally extended and primitive actions.

Interpretation. Most RL algorithms rely on local, single-step exploration approaches. Wayfarer mixes temporally extended exploration (options), designed to take the agent to the states that seem far away from the agent, with local exploration (noisy networks) designed to provide coverage of the states around the agent's location. The two heatmaps suggest that noisy networks can often flip an action selection from primitive actions to options (and vice versa), supporting the agent's exploration. Once a new room has been discovered, the agent develops a value preference for options over primitive actions. These options then support the agent in distilling a fine-grained policy over its primitive actions to consistently reach that milestone. Not only is this consistent with options accelerating credit assignment, but it also highlights a key advantage of learning option values: the agent can choose to use them if they help and ignore them if they do not.

D WAYFARER'S HYPERPARAMETERS

Table 1: Wayfarer's hyperparameters in the *Arcade Learning Environment* [63].

Symbol	Value	Meaning
$d,\ K$	$6,\ 2(d-1)=10$	number of eigenfunctions, eigenoptions
$\gamma,\ n$	$0.99,\ 3$	discount factor, multi-step returns n
$N,\ N',\ N_{\text{sel}}$	$64,\ 64,\ 32$	IQN online, target, selection quantiles
$m_\tau,\ \kappa$	$64,\ 1$	cosine embedding dims, Huber threshold
$B,\ B_{\text{rep}}$	$32,\ 512$	main & skill minibatch, rep & aux minibatch
	Adam [71]	optimiser
$\eta_\theta = \eta_\omega,\ \epsilon_{\text{Adam}}$	$5\times10^{-5},\ 3.125\times10^{-4}$	main & skill IQN step size, Adam ϵ
$\eta_\phi = \eta_\chi = \eta_{\boldsymbol{\beta}}$	$3\times10^{-4}\to3\times10^{-5}$	Laplacian encoder, inverse-dynamics & dual step size
$\boldsymbol{\beta}_0,\ b$	$-2.0,\ 0.5$	ALLO dual initialisation, barrier coefficient (fixed)
clip	30	global-norm gradient clip (all optimisers)
σ_0	0.1	NoisyNet initial noise scale
$\alpha,\ \beta_{\text{PER}},\ \epsilon_{\text{PER}}$	$0.2,\ 0.4\to1.0,\ 10^{-10}$	PER exponent, IS exponent over training, priority offset
	proportional	PER variant
P_{term}	0.2	option termination probability
w_{aux}	1.0	inverse-dynamics auxiliary weight
γ_{rep}	0.9	geometric discounting in $P(\Delta)$
update period	4 env steps	one gradient step per network
target period	10,000 env steps	$=$ 2,500 gradient steps
total steps	5×10^7 env steps	$=$ 200M frames per training run
max ep. steps	27,000 env steps	$=$ 108,000 frames at frameskip 4 ($\approx$ 30 min)
$\lvert\mathcal{D}\rvert,\ D_{\text{min}}$	$10^6,\ 20{,}000$	replay buffer capacity, min. replay samples before training

E BASELINES

Wayfarer is a *single-stream* agent: one learner, acting in one instance of the environment, learning online from the experience it generates. Rainbow and IQN provide established single-stream, model-free baselines whose components underlie Wayfarer. DreamerV3 [65] is a state-of-the-art reference beyond the model-free class. Dreamer learns from imagined experience and collects data from parallel environment instances, so this comparison places Wayfarer's performance in a broader context while differing in architecture and paradigm.

Data source. The data for Rainbow, IQN, and DQN come from *DQN Zoo* [72], a collection of reference implementations published by Google DeepMind. The data for DreamerV3 come from the original publication [65]. The DQN Zoo agents deviate from our training protocol in a few environment settings: the model-free baselines use minimal action sets, no-op starts without sticky actions, and life information. Wayfarer trains with sticky actions, the full action set, and no access to life information beyond the observations. These differences may favour the baselines.

Ablated configurations. We selected the hyperparameters for our ablated configurations of Wayfarer as follows. For *no options*, we tuned the NoisyNet initial noise level on the challenging set. For *no noisy networks* and *no option values*, the two configurations that use ϵ-greedy, we took the ϵ-decay schedule from Dabney et al. [73]: the parameter follows a linear decay from 1.0 to 0.01 over the first 4M frames and remains constant after that. We tuned *no agent-centric* on Montezuma's Revenge by evaluating the top 15 Wayfarer configurations without agent-centric regularisation. In the end, we used the same configuration as Wayfarer (minus the regularisation). Where applicable, we used the same option termination probability as Wayfarer (Appendix D).

F RESULTS

We measure mean episodic returns during training, without separate evaluation phases. Each episode has a time limit corresponding to 30 minutes of gameplay. For the main experiments, we report five runs per game for 200M frames; for the ablations, we report three runs per game for 100M frames. Each run contributes its mean episodic return over the final 10% of training to the aggregate results.

Score aggregation. We report *human-normalised scores* [41]: zero corresponds to the performance of a random agent and one to the human reference reported by Schrittwieser et al. [74]. Following Agarwal et al. [67], we summarise performance using the median, interquartile mean (IQM), mean, and optimality gap, with 95% bootstrapped confidence intervals produced using *rliable* [67].

For the main comparisons, we report results on ten games with five runs per game, except for DreamerV3, for which we use four—the minimum number of published runs available across games. Using the same number of runs per game within each method keeps games equally represented when pooling scores. For DreamerV3, we took the first four seeds in the dataset.

Area under the curve. An alternative metric for reporting results in the Arcade Learning Environment is the area under the learning curve (AUC). When we use AUC over the 200M frames, IQN and Rainbow flip places in the ranking. Wayfarer's relative performance improves according to the mean, and it now performs comparably to or better than DreamerV3 on every metric (Figure 12).

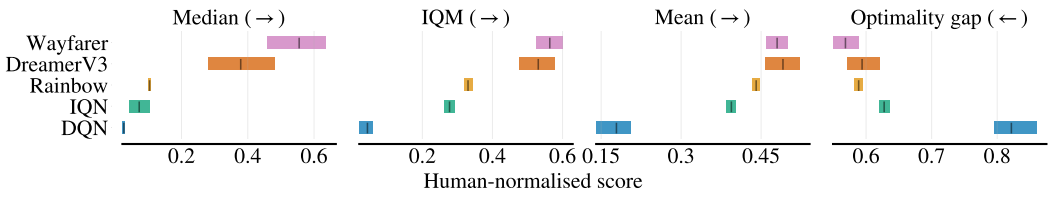


Figure 12: Aggregate metrics on the Atari 2600 challenging set, over 200M frames, with 95% CIs, using AUC of mean episodic returns over the full training period instead of the mean over last 10%. Higher mean, median, and IQM and lower optimality gap are better.

Sample efficiency. We plot IQM scores as a function of training frames (Figure 13).

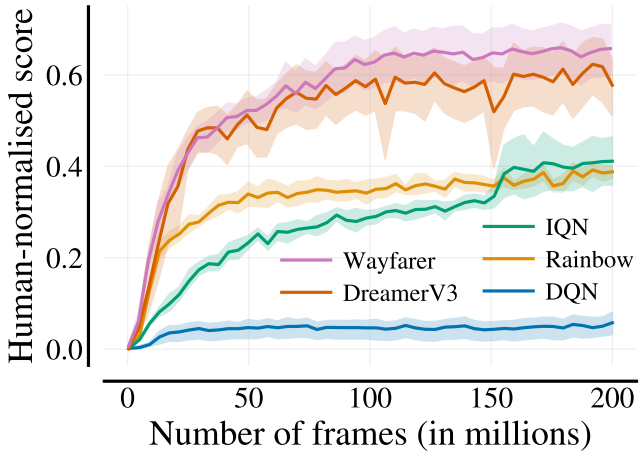


Figure 13: Interquartile mean (IQM) human-normalised scores (HNS) with 95% bootstrapped CIs.

F.1 Per-game training curves: Wayfarer vs. baselines

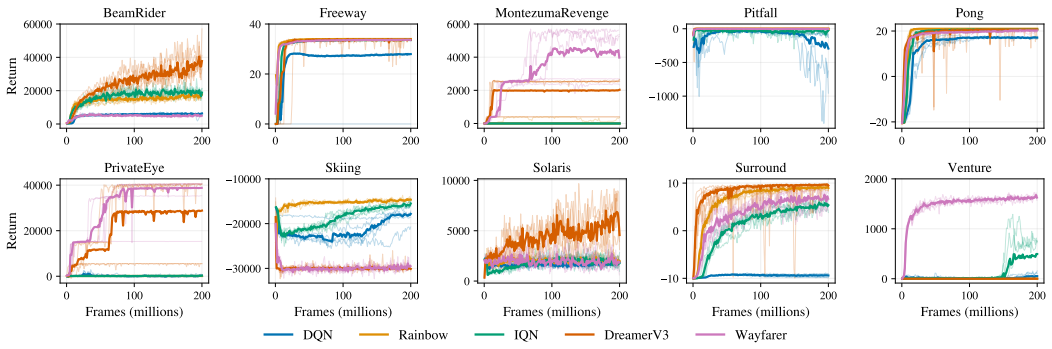


Figure 14: Mean episodic returns per game over 200M frames. The individual runs are plotted faintly, with a darker line showing the IQM across seeds (5 per game for all methods except DreamerV3, which has 4–6).

F.2 Per-game training curves: Wayfarer vs. ablations

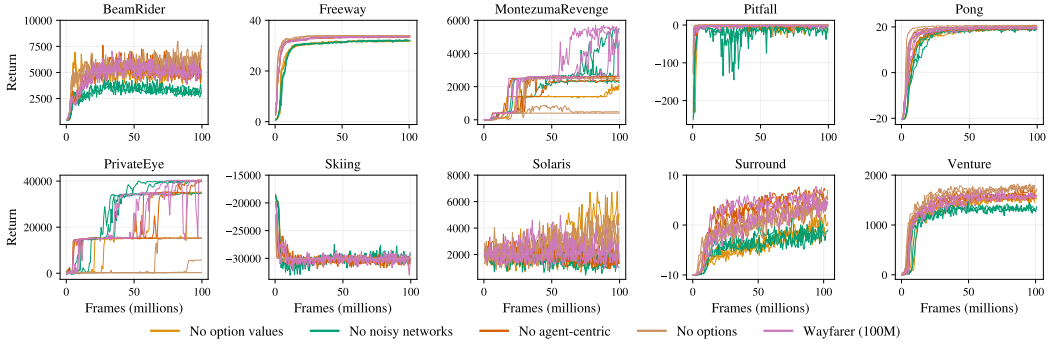


Figure 15: Mean episodic returns per game over 100M frames. The individual runs are plotted as separate lines (3 per configuration per game).

F.3 TABLES OF SCORES

Table 2: Mean episodic returns from the last 10% of training: **median** across all available seeds (5 per game for all methods except DreamerV3, which has 4–6).

Task	Random	Human	DQN	Rainbow	IQN	DreamerV3	Wayfarer
Beam Rider	363.9	16926.5	6044.7	17143.4	18865.5	33863.5	5168.2
Freeway	0.0	29.6	27.9	34.0	33.6	33.5	33.7
Montezuma	0.0	4753.3	0.0	0.0	0.0	2524.0	5107.9
Pitfall	-229.4	6463.7	-86.2	0.0	-41.0	-0.1	-2.5
Pong	-20.7	14.6	17.2	21.0	20.6	20.5	20.1
Private Eye	24.9	69571.3	121.8	100.0	171.8	39990.2	40468.4
Skiing	-17098.1	-4336.9	-17933.7	-14733.7	-16085.0	-29923.6	-29730.2
Solaris	1236.3	12326.7	1734.3	2111.1	1867.4	5858.7	1869.9
Surround	-10.0	6.5	-9.2	9.1	5.3	9.5	7.1
Venture	0.0	1187.5	31.1	0.0	692.2	0.0	1625.2

Table 3: Mean episodic returns from the last 10% of training: **IQM** across all available seeds (5 per game for all methods except DreamerV3, which has 4–6).

Task	Random	Human	DQN	Rainbow	IQN	DreamerV3	Wayfarer
Beam Rider	363.9	16926.5	6059.9	17023.1	18890.5	34681.6	5086.0
Freeway	0.0	29.6	27.9	34.0	33.6	33.5	33.7
Montezuma	0.0	4753.3	0.0	0.0	0.0	1998.2	4326.7
Pitfall	-229.4	6463.7	-245.6	0.0	-41.7	-0.1	-2.9
Pong	-20.7	14.6	17.1	21.0	20.6	20.5	20.2
Private Eye	24.9	69571.3	119.4	100.0	171.2	28538.7	38749.9
Skiing	-17098.1	-4336.9	-18060.3	-14708.0	-15905.6	-29957.3	-29723.9
Solaris	1236.3	12326.7	1753.6	2091.4	1931.6	5858.7	1933.6
Surround	-10.0	6.5	-9.4	9.0	5.4	9.5	6.6
Venture	0.0	1187.5	48.4	0.0	471.8	0.0	1622.5

Table 4: Mean episodic returns from the last 10% of training: **mean** across all available seeds (5 per game for all methods except DreamerV3, which has 4–6).

Task	Random	Human	DQN	Rainbow	IQN	DreamerV3	Wayfarer
Beam Rider	363.9	16926.5	5882.4	16930.2	18829.5	35640.4	5077.1
Freeway	0.0	29.6	22.4	34.0	33.6	33.4	33.7
Montezuma	0.0	4753.3	0.0	39.7	0.0	1828.8	4186.9
Pitfall	-229.4	6463.7	-338.1	0.0	-41.8	-0.1	-3.1
Pong	-20.7	14.6	17.0	21.0	20.6	20.4	20.1
Private Eye	24.9	69571.3	132.2	159.9	179.6	26287.2	34418.4
Skiing	-17098.1	-4336.9	-18482.4	-14730.3	-15882.9	-29960.7	-29708.8
Solaris	1236.3	12326.7	1745.9	2055.0	1933.8	5829.3	1935.4
Surround	-10.0	6.5	-9.4	9.0	5.4	9.5	6.7
Venture	0.0	1187.5	50.4	0.0	435.3	0.0	1618.9

G WHAT DOES THE AGENT LEARN TO REPRESENT?

In the game Freeway, the objective is to cross the road as many times as possible, while avoiding cars, within a limited amount of time. The agent can only move up or down. Figure 16 visualises the smoothest non-constant eigenfunction learned with Laplacian representation learning directly

from observations versus that learned from a contingent-aware feature space. It shows the problem described in Section 3.3 in action: without agent-centric regularisation, the two options corresponding to the visualised smoothest non-constant eigenfunction could receive intrinsic rewards driven by changes outside the agent's control (it cannot affect the motion of the cars). With regularisation, the eigenfunction instead appears to track the agent's vertical position: the agent can only move up or down along the vertical path.

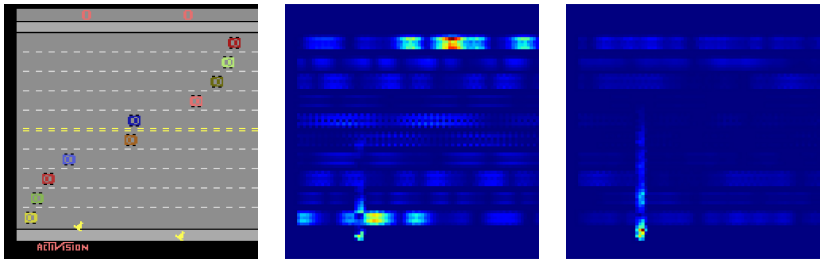


Figure 16: Freeway. *(Left)*. An RGB frame from the game. *(Center)* The smoothest non-constant eigenfunction tracks exogenous structure—the two slowest cars—whose motion is unaffected by the agent. *(Right)* With agent-centric regularisation, the smoothest non-constant eigenfunction tracks only the agent's vertical position, which is the part of the observation under the agent's control.

Figure 17 provides a lens on the representation learning in Montezuma's Revenge. The mechanics of the game are explained in Appendix C.1, alongside a map of the first level to provide some spatial context (Figure 10). We visualise which parts of the agent's observations, in three of the first rooms, the representation is most sensitive to. The representation appears to track features associated with the environment's *bottlenecks*. Bottlenecks, by definition, gate transitions between weakly connected regions in an environment [75]. The smoothest non-constant eigenfunctions tend to spike between such regions [51, Figure 4, p. 11], so options derived from them may traverse such bottlenecks.

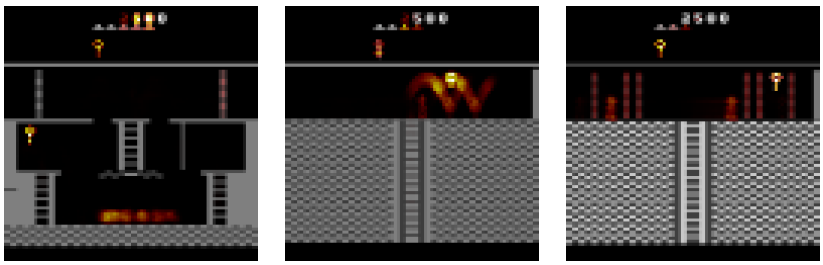


Figure 17: Montezuma's Revenge (rooms 1, 2, and 7). The learned Laplacian representation is sensitive to changes in pixels associated with progress, including keys, doors, skulls, laser gates, and parts of the heads-up display. When these features change, the behaviours that can lead to further progress may also change, making them plausible indicators of transitions between weakly connected regions of the experienced state space.

G.1 Feature attribution

We analyse the learned Laplacian eigenfunctions using the basis-invariant attribution method of Hilton [76]. The method can be viewed as a *sensitivity-weighted analogue of PCA*: it identifies directions of variation in the observations that matter to a given eigenfunction, accounting for both their variance in the input space and the eigenfunction's sensitivity to them.

Given a fixed representation network checkpoint, let $u_i : \mathbb{R}^m \to \mathbb{R}$ denote the i-th learned eigenfunction, where m is the dimensionality of a flattened observation $\mathbf{x} \in \mathbb{R}^m$. Let $\mathcal{D}_{\text{attr}}$ denote the dataset of observations used for feature attribution; we write $\mathbb{E}_{\mathcal{D}_{\text{attr}}}$ for an empirical average over these observations.

Define the observation mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, and the uncentred gradient covariance matrix $\boldsymbol{\Gamma}_i$, by

$$\boldsymbol{\mu} := \mathbb{E}_{\mathcal{D}_{\text{attr}}}[\mathbf{x}],$$
$$\boldsymbol{\Sigma} := \mathbb{E}_{\mathcal{D}_{\text{attr}}}\left[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top\right],$$
$$\boldsymbol{\Gamma}_i := \mathbb{E}_{\mathcal{D}_{\text{attr}}}\left[\nabla_{\mathbf{x}} u_i(\mathbf{x}) \nabla_{\mathbf{x}} u_i(\mathbf{x})^\top\right],$$

where $\nabla_{\mathbf{x}}$ denotes the gradient with respect to the observation $\mathbf{x}$.

The method seeks a projection matrix $\mathbf{P} \in \mathbb{R}^{m\times m}$ satisfying $\mathbf{P}^2 = \mathbf{P}$ and $\operatorname{rank}(\mathbf{P}) \le k$, where k is the maximum number of directions we wish to retain. Following the derivation in Hilton [76], we approximately minimise the surrogate objective

$$\mathcal{L}_{\text{attr},i}(\mathbf{P}) := \operatorname{Tr}\big((\boldsymbol{I}_m - \mathbf{P})\boldsymbol{\Sigma}(\boldsymbol{I}_m - \mathbf{P})^\top \boldsymbol{\Gamma}_i\big),$$

where $\boldsymbol{I}_m$ is the $m \times m$ identity matrix and Tr denotes the trace.

The matrix $\boldsymbol{\Gamma}_i$ weights reconstruction errors by the sensitivity of u_i. When $\boldsymbol{\Gamma}_i = \boldsymbol{I}_m$, the objective reduces to the PCA reconstruction objective.

We use the fast approximate algorithm described in Section 4.1 of Hilton [76], based on randomised subspace iteration [77]. The algorithm returns a matrix $\mathbf{P}_A$ whose columns form a basis for the image of $\mathbf{P}$. We reshape each column of $\mathbf{P}_A$ to the observation dimensions to obtain an attribution map.

Applying the method to Atari 2600 games. We use observations stored in experience replay buffers saved alongside the network checkpoints. Figure 16 shows two maps from the game Freeway, each computed separately for the smoothest non-constant eigenfunction from the corresponding training run. For Montezuma's Revenge, we used a vector-valued extension of the method [76, Section 4.4]. For an observation $\mathbf{x} \in \mathbb{R}^m$, the representation network outputs $\mathbf{u} = f_\phi\big(f_\psi(\mathbf{x})\big) \in \mathbb{R}^d$, where d is the number of learned eigenfunctions, f_ψ is the shared encoder, and f_ϕ is the Laplacian head (Section 3.3). Then, let

$$\mathbf{J}_{\mathbf{u}}(\mathbf{x}) := \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \in \mathbb{R}^{d\times m}$$

denote the Jacobian of this output with respect to the observation. The vector-valued extension replaces $\boldsymbol{\Gamma}_i$ with

$$\boldsymbol{\Gamma} := \mathbb{E}_{\mathcal{D}_{\text{attr}}}\left[\mathbf{J}_{\mathbf{u}}(\mathbf{x})^\top \mathbf{J}_{\mathbf{u}}(\mathbf{x})\right].$$

This gives one set of directions for the Laplacian representation as a whole. We partitioned the saved observations by room and, for each room, used the resulting subsample as $\mathcal{D}_{\text{attr}}$ to compute a separate $\mathbf{P}_A$. One such attribution map is shown in the top-right panel of Figure 1. In Figure 17, the maps are overlaid on room templates formed by taking the pixel-wise median of observations from the corresponding room.

H Extended related work

Wayfarer builds on representation-driven option discovery, on Laplacian and agent-centric representation learning, and on the broader option-discovery literature. As such, we discuss how Wayfarer relates to existing approaches in the literature.

H.1 Learning and using Laplacian-based options online

Like Wayfarer, Deep Covering Eigenoptions (DCEO) [26] learns the smoothest eigenfunctions of the Laplacian online, from high-dimensional observations, and derives options from them, instantiating the representation-driven option discovery (ROD) cycle [51, 52]. DCEO, however, uses its options only for exploration: when its ϵ-greedy policy explores, the agent follows a randomly chosen option until termination. Indeed, Klissarov & Machado [26] explicitly identify learning option values as a direction for future work (p. 9). Wayfarer's main network instead learns task-reward values over primitive actions and options alike, and chooses among them by their estimated return.

Wayfarer also differs in other ways: it regularises its representation with multi-step inverse dynamics (Section 3.3), learns the representation with ALLO [33] rather than GGDO [19], uses a unified skill network to learn the options, and uses IQN [50] rather than C51 [48] with Rainbow.

Learning option values introduces difficulties of its own. Kotamreddy & Machado [68] show that, with exploration controlled for, learning eigenoption values can accelerate credit assignment in tabular settings. However, they struggled to show gains when learning online or with function approximation. Wayfarer learns option values in exactly this setting: online, with deep neural networks, from high-dimensional observations. Our ablations separate the effect of learning option values from that of the exploration mechanism, and include a DCEO-style variant that uses the options only for exploration (Section 4).

Novel Exploration via Orthogonality (NEO) [20] also discovers options online and learns their values from task reward, but in tabular domains. NEO adds visitation counts to a modified Laplacian, so that its eigenvectors, and the option policies defined from them, lead towards novel parts of the state space. NEO learns option values with SMDP Q-learning over entire option executions. The algorithm computes the eigenvectors of this modified Laplacian, with option policies defined over the eigenvectors. Wayfarer instead learns both its representation and its option policies from high-dimensional observations, and its main network updates option values with the same n-step target as primitive actions regardless of option duration (Section 3.1).

Hierarchical Novel Eigen Options (HNEO) [78] extends NEO to pixel-based domains. They do this by approximating counts and using this as a reward bonus on top of the rewards given by the Laplacian eigenfunctions. Like NEO, HNEO learns option values with SMDP updates. HNEO builds on PQN [79] and learns from many parallel instances of the environment, whereas Wayfarer learns from a single stream of experience. The two methods also regularise Laplacian representation learning differently: HNEO towards novelty, Wayfarer towards agent-centric features.

Earlier work established several parts of Wayfarer's approach to ROD. Machado et al. [16] derived Laplacian-based options from raw pixels, learning a deep successor representation [80] from a fixed dataset collected by a uniformly random policy. Deep covering options [18] scale covering options [17] to large state spaces: each option terminates in an under-explored region identified by an approximate Laplacian eigenfunction [32] and joins the action set of a higher-level policy that learns its value from task reward. Options are added one at a time, and the method was evaluated quantitatively in low-dimensional state spaces, with qualitative results in Atari. Wayfarer likewise values its options with task reward, but does not only rely on the smoothest (non-constant) eigenfunction and learns all options simultaneously in one online loop from high-dimensional experience.

Continual curiosity-driven skill acquisition (CCSA) [81] was among the first methods to couple perceptual abstraction and skill acquisition from raw pixels, storing and reusing acquired skills to support further skill discovery. CCSA learns slow features, which have been shown to be mathematically related to Laplacian eigenmaps [62], and intrinsically rewards transitions that produce large changes in their outputs. CCSA learns each skill by model-based policy iteration over a transition model of abstract states obtained by clustering the outputs of the slow features. Wayfarer's options likewise seek change in slowly varying features, but each is rewarded for moving one eigenfunction in one direction, and all are discovered without a learned transition model using deep Q-learning.

H.2 Unsupervised skill discovery for downstream tasks

Unsupervised skill discovery asks how an agent can acquire useful behaviours without task rewards. Variational intrinsic control [54], for example, maximises mutual information between options and their termination states, encouraging behaviours with distinguishable outcomes. Metric-Aware Abstraction (METRA) [82] grounds skill discovery in a temporal-distance metric and scales to locomotion from pixels. Zheng et al. [83] analyse METRA through the lens of mutual-information skill learning, derive a connection between its representation learning objective and contrastive learning, and introduce Contrastive Successor Features (CSF). In their hierarchical experiments, METRA and CSF first pre-train skills and then train a controller whose actions are those skills. Wayfarer does not separate these phases: its task-directed choices shape the experience from which it learns its representation and options, and the changing options in turn shape the behaviours available to it.

H.3 Spectral and agent-centric representations

ROD requires a representation that remains useful as the agent's experience changes, yet spectral graph-drawing objectives such as ALLO [33] are typically trained under a fixed, often uniformly random, policy [34]. Erraqabi et al. [84] observe that the temporally contrastive formulation of the Laplacian assumes uniform access to all states, and pair representation learning with a skill-based covering policy that improves the training distribution. In Wayfarer, the options that broaden this distribution are those the main network selects for task return (as well as those it explores with) and not a separate covering policy. Ahmed et al. [34] study Laplacian representation learning under a changing policy: they introduce the Asymmetric Graph Drawing Objective (AGDO), prove that its online optimisation converges when policy drift is bounded, and find that accuracy suffers when it is not. Wayfarer uses ALLO, but holds its penalty coefficient fixed and applies a symmetric log transformation to its outputs to stabilise learning from changing experience (Section 3.3). Combining Wayfarer with an objective like AGDO is left for future work.

Spectral structure can also drive exploration without options: the Spectral Bellman Method [85] learns state-action features aligned with the structure of Bellman updates and explores by Thompson sampling in the learned feature space. In comparison, Wayfarer uses its Laplacian eigenfunctions to define rewards for temporally extended behaviours.

The source of temporal variation also matters when a representation defines rewards for behaviour. A timer can change slowly regardless of the agent's actions, providing a predictable feature without providing a useful objective for an option. *Contingency awareness* [28] captures the distinction between changes influenced by the agent and those determined by the environment, and has been used to construct features for value approximation in Atari. More recent work formalises the problem of extracting endogenous state information in the presence of exogenous distractors, developing methods based on multistep inverse dynamics prediction [30, 31, 69]. Single-step inverse dynamics has also been used to learn the feature space in which prediction-error intrinsic rewards are computed [86], and to focus novelty on controllable aspects of Atari observations [29, 59]. Wayfarer regularises the encoder shared with its Laplacian head by multi-step inverse-dynamics prediction, which encourages the representation network to encode information from which the agent's actions can be inferred, thereby shaping the eigenfunctions that define option rewards. The regulariser supplies an inductive bias and our experiments examine its effects (Section 4, Appendix G).

Empowerment offers another link between skill learning and control-relevant representations: Bastankhah et al. [87] show that agents maximising empowerment induce representations invariant to control-irrelevant features. Wayfarer pursues a related aim through inverse-dynamics regularisation of a spectral representation.

H.4 Abstractions from task rewards and subgoals

Other methods derive their option-discovery objectives from task rewards or from designer-specified subgoals. The option-critic architecture [11] learns option policies and termination functions directly from task reward. FeUdal Networks (FuN) [10] train a manager to set goals in a learned latent space and a worker to pursue them. The worker receives an intrinsic reward for following the goals, but the manager learns from task reward which goals to set. In both methods, task reward determines which behaviours are learned, but where non-zero rewards are sparse such methods grounded in reward have no signal to build on. Wayfarer's representation-derived rewards, in contrast, shape its options before the agent encounters informative task rewards.

Hierarchical DQN (h-DQN) [13], instead of using task reward, learns lower-level policies for designer-given subgoals, as well as a higher-level policy that selects among them. Similarly, earlier intrinsically motivated RL organised skill acquisition around designer-specified "salient" events [12]. These methods show how to learn useful behaviours once their objectives are given, whereas Wayfarer derives those objectives from interaction with the environment.

Task reward can also enter through the representation itself. Tse et al. [88] study the default representation, an analogue of the successor representation that also accounts for the problem's reward dynamics, and show that it gives rise to reward-aware behaviour in option discovery. Wayfarer keeps the representation reward-free: task reward enters only through the main network, which values the options and, through its choices, shapes the experience from which they are learned.

Bagaria et al. [89] use feature attribution to identify a small subset of relevant features for each subgoal, and learn options that reach the resulting abstract subgoals. Wayfarer uses feature attribution only to analyse its learned eigenfunctions (Appendix G), and feature attribution plays no part in its option rewards or termination.

Skill-driven neurosymbolic state abstractions [90] reverse the dependence at the core of ROD: rather than deriving options from a representation, they construct abstract states compatible with a given set of abstract actions. It is an interesting complementary approach to Wayfarer; we leave thoughts about fusing the approaches for future work.

I Normalisation and effective step size

We observed that Wayfarer can plateau early in games requiring continual exploration, and we suspect that implicit step size decay contributes to these plateaus. With LayerNorm [91], used to stabilise deep Q-learning [79], growing weight norms implicitly reduce the effective step size thus slowing adaptation to new experience [92, pp. 4–5]. We experimented with Normalize-and-Project (NaP) [92], a simple modification that fixes weight norms in normalised layers to prevent this implicit decay. With the effective step size held constant, Wayfarer achieved a mean episodic return of over 7,000 on Montezuma's Revenge within the 200M frame budget (Figure 18).[△] This result motivates further work on sustaining plasticity through mechanisms such as NaP.

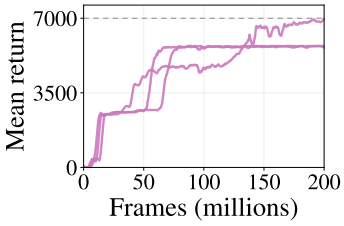


Figure 18: Mean episodic return in Montezuma's Revenge with NaP and a constant effective step size.