Unsupervised Skill-Discovery and Skill-Learning in Minecraft

Anonymous Authors[1]

Abstract

Pre-training Reinforcement Learning agents in a task-agnostic manner has shown promising results. However, previous works still struggle in learning and discovering meaningful skills in high-dimensional state-spaces, such as pixel-spaces. We approach the problem by leveraging unsupervised skill discovery and self-supervised learning of state representations. In our work, we learn a compact latent representation by making use of variational and contrastive techniques. We demonstrate that both enable RL agents to learn a set of basic navigation skills by maximizing an information theoretic objective. We assess our method in Minecraft 3D pixel maps with different complexities. Our results show that representations and conditioned policies learned from pixels are enough for toy examples, but do not scale to realistic and complex maps. To overcome these limitations, we explore alternative input observations such as the relative position of the agent along with the raw pixels.

1. Introduction

Reinforcement Learning (RL) (Sutton and Barto, 2018) has witnessed a wave of outstanding works in the last decade, with special focus on games (Schrittwieser et al. (2020), Vinyals et al. (2019), Berner et al. (2019)), but also in robotics (Akkaya et al. (2019), Hwangbo et al. (2017)). In general, these works follow the classic RL paradigm where an agent interacts with an environment performing some action, and in response it receives a reward. These agents are optimized to maximize the expected sum of future rewards.

Rewards are usually handcrafted or overparametrized, and this fact becomes a bottleneck that prevents RL to scale. For this reason, there has been an increasing interest in training agents in a task-agnostic manner during the last few years, making use of intrinsic motivations and unsupervised techniques. Recent works have explored the unsupervised learning paradigm (Campos et al. (2020); Gregor et al. (2017); Eysenbach et al. (2018); Warde-Farley et al. (2018); Burda et al. (2018); Pathak et al. (2017)), but RL is still far from the remarkable results obtained in other domains. For instance, in computer vision, Chen et al. (2021) achieve an 81% accuracy on ImageNet training in a self-supervised manner, or Caron et al. (2021) achieves state-of-the-art results in image and video object segmentation using Visual Transformers (Dosovitskiy et al., 2021) and no labels at all. Also, in natural language processing pre-trained language models such as GPT-3 (Brown et al., 2020) have become the basis for other downstream tasks.

Humans and animals are sometimes guided through the process of learning. We have good priors that allow us to properly explore our surroundings, which leads to discovering new skills. For machines, learning skills in a task-agnostic manner has proved to be challenging (Warde-Farley et al., 2018; Lee et al., 2020). These works state that training pixel-based RL agents end-to-end is not efficient because learning a good state representation is unfeasible due to the high dimensionality of the observations. Moreover, most of the successes in RL come from training agents during thousands of simulated years (Berner et al. (2019)) or millions of games (Vinyals et al. (2019)). This learning approach is very sample inefficient and sometimes limits its research because of the high computational budget it may imply. As a response, some benchmarks have been proposed to promote the development of algorithms that can reduce the number of samples needed to solve complex tasks. This is the case of MineRL (Guss et al. (2021)) or ProcGen Benchmark (Cobbe et al. (2020)).

Our work is inspired by Campos et al. (2020) and their *Explore, Discover and Learn* (EDL) paradigm. EDL relies on *empowerment* (Salge et al., 2014) for motivating an agent intrinsically. Empowerment aims to maximize the influence of the agent over the environment while discovering novel skills. As stated by Salge et al. (2014), this can be achieved my maximizing the mutual information between sequences of actions and final states. Gregor et al. (2017) introduces a novel approach that instead of blindly committing to a sequence of actions, each action depends on the observation from the environment. This is achieved by maximizing the

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

mutual information between inputs and some latent variables. Campos et al. (2020) embraces this approach as we do. However, the implementation by Campos et al. (2020) makes some assumptions that are not realistic for pixel observations. Due to the Gaussian assumption at the output of variational approaches, the intrinsic reward is computed as the reconstruction error, and in the pixel domain this metric does not necessarily match the distance in the environment space. Therefore, we look for alternatives that suit our requirements: we derive a different reward from the mutual information, and we study alternatives to the variational approach.

This work focuses on learning meaningful representations, discovering skills and training latent-conditioned policies. In any of the cases, our methodology does not require any supervision and works directly from pixel observations. Additionally, we also study the impact of extra input information in the form of position coordinates. Our proposal is tested over the MineRL (Guss et al., 2021) environment, which is based on the popular Minecraft videogame. Even though the game proposes a final goal, Minecraft is well known by the freedom that it gives to the players, and actually most human players use this freedom to explore this virtual world following their intrinsic motivations. Similarly, we aim at discovering skills in Minecraft without any extrinsic reward.

We generate random trajectories in Minecraft maps with little exploratory challenges, and also study contrastive alternatives that exploit the temporal information throughout a trajectory. The contrastive approach aims at learning an embedding space where observations that are close in time are also close in the embedding space. A similar result can be achieved by leveraging the agents' relative position in the form of coordinates. In the latter, the objective is to infer skills that do not fully rely on pixel resemblance, but also take into account temporal and spatial relationships.

Our final goal is to discover and learn skills that can be potentially used in more broad and complex tasks. Either by transferring the policy knowledge or by using hierarchical approaches. Some works have already assessed this idea specially in robotics (Florensa et al., 2017) or 2D games (Campos et al., 2021). Once the pre-training stage is completed and the agent has learned some basic behaviours or skills, the agent is exposed to an extrinsic reward. These works show how the agents leverage the skill knowledge to learn much faster and encourage proper exploration of the environment in unrelated downstream tasks. However, transferring the policy knowledge is not as straightforward as in other deep learning tasks. If one wants to transfer behaviours (policies), the change in the task might lead to catastrophic forgetting.

Our contributions are the following:

- We demonstrate that variational techniques are not the only ones capable of maximizing the mutual information between inputs and latent variables by leveraging contrastive techniques.
- We provide alternatives for discovering and learning skills in procedurally generated maps by leveraging the agents coordinate information.
- We succesfully implement the reverse form of the mutual information for optimizing pixel-based agents in a complex 3D environment.

2. Related Work

Intrinsic Motivations (IM) are very helpful mechanisms to deal with sparse rewards. In some environments the extrinsic rewards are very difficult to obtain and, therefore, the agent does not receive any feedback to progress. In order to drive the learning process without supervision, we can derive intrinsic motivations as proxy rewards that guide the agents towards the extrinsic reward or just towards better exploration.

Skill Discovery. We relate Intrinsic Motivations to the concept of *empowerment* (Salge et al., 2014), a RL paradigm in which the agent looks for the states where it has more control over the environment. Mohamed and Rezende (2015) derived a variational lower bound on the mutual information that allows to maximize empowerment. Skill discovery extends this idea from the action level to temporally-extended actions. Florensa et al. (2017) merges skill discovery and hierarchical architectures. They learn a high-level policy on top of some basic skills learned in a task-agnostic way. They show how this set-up improves the exploration and enables faster training in downstream tasks. Similarly, Achiam et al. (2018) emphasize in learning the skills dynamically using a curriculum learning approach, allowing the method to learn up to a hundred of skills. Instead of maximizing the mutual information between states and skills they use skills and whole trajectories. Eysenbach et al. (2018) demonstrates that learned skills can serve as an effective pretraining mechanism for robotics. Our work follows their approach regarding the use of a categorical and uniform prior over the latent variables. Campos et al. (2020) exposes the lack of coverage of previous works. They propose *Explore, Discover and Learn* (EDL), a method for skill discovery that breaks the dependency on the distributions induced by the policy. Warde-Farley et al. (2018) provides an algorithm for learning goal-conditioned policies using an imitator and a teacher. They demonstrate the effectiveness of their approach in pixel-based environments like Atari, DeepMind Control Suite and DeepMind Lab.

Intrinsic curiosity. In a more broad spectrum we find methods that leverage intrinsic rewards that encourage ex-

ploratory behaviours. Pathak et al. (2017) present an *Intrinsic Curiosity Module* that defines the curiosity or reward as the error predicting the consequence of its own actions in a visual feature space. Similarly, Burda et al. (2018) uses a Siamese network where one of the encoder tries to predict the output of the other. The bonus reward is computed as the error between the prediction and the random one.

Goal-oriented RL. Many of the works dealing with skill-discovery end up parameterizing a policy. This policy is usually conditioned in some goal or latent variable $z \sim Z$. The goal-conditioned policy formulation along with function approximation was introduced by Schaul et al. *Hindsight Experience Replay* (HER) by Andrychowicz et al. (2017) allows sample-efficient learning from environments with sparse rewards. HER assumes that any state can be a goal state. Therefore, leveraging this idea, the agent learns from failed trajectories as if the final state achieved was the goal state. Trott et al. (2019) proposes to use pairs of trajectories that encourage progress towards a goal while learns to avoid local optima. Among different environments, *Sibling Rivalry* is evaluated in a 3D construction task in Minecraft. The goal of the agent is to build a structure by placing and removing blocks. They use the number of block-wise differences as a naive shaped reward which causes the agent to avoid placing blocks. Simply by adding Sibling Rivalry they manage to improve the degree of construction accuracy.

Representation Learning in RL. In RL we seek for low-dimensional state representations that preserve all the information and variability of the state space in order to make decisions that eventually maximize the reward. This becomes crucial when dealing with pixel-based environments. Ghosh et al. (2018) aims to capture those factors of variation that are important for decision making and are aware of the dynamics of the environment without the need for explicit reconstruction. Lee et al. (2020) also makes use of variational inference techniques for estimating the posterior distribution, but breaks the Markovian assumption by conditioning the probability of the latent variables with past observations. Oord et al. (2018) leverages powerful autoregressive models and negative sampling to learn a latent space from high-dimensional data. This work introduces the InfoNCE loss based on Noise Contrastive Estimation. The intuition behind is that we learn a latent space that allows to classify correctly positive pairs while discriminating from negative samples. We make use of this loss in our contrastive experiments, as well as the following works explained. Laskin et al. (2020) trains an end-to-end model by performing off-policy learning on top of extracted features. This features are computed using a contrastive approach based on pairs of augmented observations, while Stooke et al. (2020) picks pairs of delayed observations. All these works learn representations from pixel-based observations, except for the first one by Ghosh et al. (2018).

3. Information-theoretic skill discovery

Intrinsic Motivations (IM) can drive the agents learning process in the absence of extrinsic rewards. With IM, the agents do not receive any feedback from the environment, but must autonomously learn the possibilities available in it. To achieve that, the agents aim to gain resources and influence on what can be done in the environment. In the *empowerment* (Salge et al., 2014) framework, an agent will look for the state in which has the most control over the environment. This concept usually deals with simple actions. In contrast, *skill-discovery* temporally abstracts these simple actions to create high-level actions that are dubbed as *skills* or *options*. Skill-discovery is formulated as maximizing the mutual information between inputs and skills. This encourages the agent to learn skills that derive as many different input sequences as possible, while avoiding overlap between sequences guided by different skills. Therefore, *skill-discovery* can ease the exploration in complex downstream tasks. Instead of executing random actions, the agent can take advantage of the learned behaviours in order to perform smarter moves towards states with potential extrinsic rewards.

In the next section we formulate the mathematical framework that is used through our work. First we define the classic Markov decision process that typically provide a mathematical formulation for reinforcement learning tasks. Later on, we introduce the tools from information-theory that allows us to define the maximization of the mutual information.

3.1. Preliminaries

Let us consider a Markov decision process (MDP) without rewards as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P})$. Where $\mathcal{S}$ is the high-dimensional state space (pixel images), $\mathcal{A}$ refers to the set of actions available in the environment and $\mathcal{P}$ defines the transition probability $p(s_{t+1}|s_t, a)$. We learn latent-conditioned policies $\pi(a|s, z)$, where the latent $z \in \mathcal{Z}$ is a random variable.

Given the property of symmetry, the mutual information ($\mathcal{I}$) can be defined using the Shannon Entropy ($\mathcal{H}$) in two ways (Gregor et al. (2017)):

$$\begin{aligned} \mathcal{I}(S, Z) &= \mathcal{H}(Z) - \mathcal{H}(Z|S) \quad \rightarrow \quad \text{reverse} \\ &= \mathcal{H}(S) - \mathcal{H}(S|Z) \quad \rightarrow \quad \text{forward} \end{aligned} \tag{1}$$

For instance, we derive the reverse form of the mutual information (MI):

$$\mathcal{I}(S, Z) = E_{s, z \sim p(z,s)}[\log p(z|s)] - E_{z \sim p(z)}[\log p(z)] \tag{2}$$

The posterior $p(z|s)$ is unknown due to the complexity to

marginalize the evidence $p(s) = \int p(s|z)p(z)dz$. Hence, we approximate it with $q_\phi(z|s)$ by using variational inference or contrastive techniques. As a result, we aim at maximizing a lower bound based on the KL divergence. For a detailed derivation we refer the reader to the original work from Mohamed and Rezende (2015).

Therefore, the MI lower bound becomes:

$$\mathcal{I}(S, Z) \geq E_{s,z\sim p(z,s)}[\log q_\phi(z|s)] - E_{z\sim p(z)}[\log p(z)] \tag{3}$$

The prior $p(z)$, can either be learned through the optimization process (Gregor et al., 2017), or fixed uniformly beforehand (Eysenbach et al., 2018). In our case, since we want to maximize the uncertainty over $p(z)$, we define a categorical uniform distribution.

3.1.1. Variational Inference

Variational inference is a well-known method for modelling posterior distributions. Its main advantage is that it avoids computing the marginal $p(s)$, which is usually impossible. Instead, one selects some tractable families of distributions q as an approximation of p.

$$p(z|s) \approx q(z|\theta) \tag{4}$$

We fit q with sample data to learn the distribution parameters θ. In particular, we make use of Variational Auto-Encoders (VAE) (Kingma and Welling (2014)) with categorical classes, namely VQ-VAE (van den Oord et al., 2017), for modelling the mappings from inputs to latents and backwards. The encoder $q_\phi(z|s)$ models $p(z|s)$ and the decoder $q_\psi(s|z)$ models $p(s|z)$.

3.1.2. Contrastive Learning

Contrastive learning is a subtype of self-supervised learning that consists in learning representations from the data by comparison. This field has quickly evolved in the recent years, especially in computer vision. Within this context, the comparisons are between pairs of images. Examples of positive pairs may be augmented versions of a given image like crops, rotations, color transformations, etc; and the negative pairs maye be the other images from the dataset. In this case, representations are learned by training deep neural networks to distinguish between positive or negative pairs of observations. The main diference between contrastive self-supervised (or unsupervised) learning and metric learning is that the former does not require any human annotation, while the later does. Learning representations with no need of labeling data allows scaling up the process and overcoming the main bottleneck when training deep neural networks: the scarcity of supervisory signals.

Similar to what is done in variational techniques, we can compute the mutual information between inputs and latents. In this case, instead of learning the latents by the reconstruction error, they are learned by modeling global features between positive and negative pairs of images. Intuitively, two distinct augmentations (positive pair) should be closer in the embedding space than two distinct images (negative pair) from the dataset. Therefore, we need two encoders in paralel instead of an encoder-decoder architecture. The second encoder is usually called momentum encoder (Chen et al. (2021)) and its weights are updated using exponential moving averages of the main encoder.

In each training step, we forward a batch of different original images through the main encoder while we forward the positive pairs of each of those images through the momentum encoder. In a batch of N samples, we have for each positive pair, $N-1$ negative pairs. We define z as the output of the encoder and z' as the output of the momentum encoder. Then, e is a convolutional neural encoder and h is a projection head (small multi-layer perceptron) that returns a latent z, ($z = h(e_\theta(s))$).

At the output of the network we can perform a categorical cross-entropy loss where the correct classes are the positive pairs in the batch. The correct classes are in the diagonal of the resulting matrix $z^T W z'$, where W is a projection matrix learned during training. In the other positions, we have the similarity between negative pairs in the embedding space. Minimizing this loss, whose name is InfoNCE (Oord et al., 2018), will encourage the model to find global features that can be found in augmented versions from an image. Moreover, as stated by their authors Oord et al. (2018), minimizing the InfoNCE Eq. 5 loss maximizes the mutual information between inputs and latents. As a result, we learn the desired mapping from input states s to latent vectors z.

$$\mathcal{L}_{InfoNCE} = \log \frac{\exp(z^T W z')}{\exp(z^T W z') + \sum_{i=0}^{K-1} \exp(z^T W z'_i)} \tag{5}$$

4. Methodology

In this section we show the implementation details adopted for each of the stages of our method.

4.1. Exploration

EDL (Explore, Discover and Learn) (Campos et al., 2020) provides empirical and theoretical analysis of the lack of coverage of some methods leveraging the mutual information for discovering skills. Either by using the forward or the reverse form of the MI, the reward given to novel states is always smaller than the one given to known states. The main problem dwells in the induced

state distributions used to maximize the mutual information. Most works reinforce already discovered behaviours since they induce the state distribution from a random policy, $p(s) \approx \rho_\pi(s) = E_z[\rho_\pi(s|z)]$. EDL is agnostic to how $p(s)$ is obtained, so we can infer it in very different ways. One manner would be to induce it by leveraging a dataset of expert trajectories that will encode human-priors that are usually learned poorly with information-theoretic objectives. Another way is to make use of an exploratory policy that is able to induce a uniform distribution over the state space. Since this is not clearly solved in complex 3D environments yet, we explore by using random policies in bounded maps.

4.2. Skill-Discovery

In Section 3, we proposed two distinct approaches for modelling the mapping from the observations s to the latent variables z: variational inference and contrastive learning. The skill-discovery pipelines for both approaches are depicted in Figure 1, and are described in the remaining of this section.

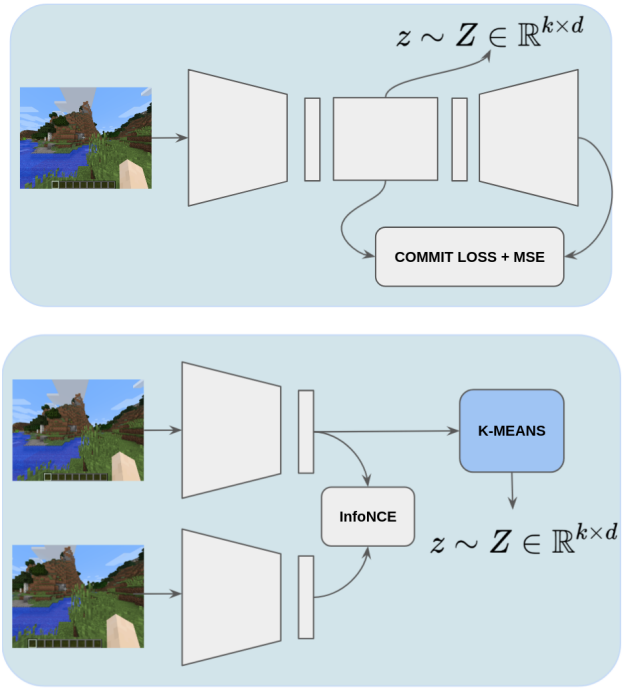


Figure 1. Skill-discovery pipeline with (top) variational and (bottom) contrastive approaches.

4.2.1. Variational Inference

Vector-Quantized VAE (VQ-VAE) (van den Oord et al., 2017) is a variational model that allows us to have a categorical distribution $p(z)$ as a bottleneck (also called codebook), an encoder q_ψ that estimates the posterior $p(z|s)$, and the decoder q_ϕ that estimates $p(s|z)$.

Before training, the model requires to fix the length of the codebook. This determines the granularity of the latent variables. If we choose a large number of codes, we will end up with latent variables that encode very similar states. Instead, if we choose a small number, we will encourage the model to find latents that generalize across diverse scenarios. The perplexity metric measures the number of codewords needed to encode our whole state distribution. In practice, this metric is computed per batch during training. Since it is not possible to know beforehand the number of useful latent variables that our model will discover, one can iterate over different codebook lengths until they find a good trade-off between generalization and granularity.

In EDL (Campos et al., 2020), the purpose of training a variational auto-encoder was to discover the latent variables that condition the policies in the Learning stage. But the learned representations were discarded. Instead, in our case, we also leverage the representations learned in the encoder and decoder for training the RL agent. This allows for faster and more efficient trainings of the RL agents.

4.2.2. Contrastive Learning

In the contrastive case, we follow the idea of Stooke et al. (2020) where the positive pairs are delayed observations from a specific trajectory. Once the latents are learned, we define a categorical distribution over latents by clustering the embedding space of the image representations (Yarats et al., 2021). This step is performed using K-Means with K clusters equal to the length of the VQ-VAE codebook.

At this point, we have the same set-up for both the contrastive and variational approaches. The K-Means centroids are equivalent to the VQ-VAE codebook embeddings, so we can maximize the mutual information between the categorical latents and the inputs with the same Equation 2.

4.3. Skill-Learning

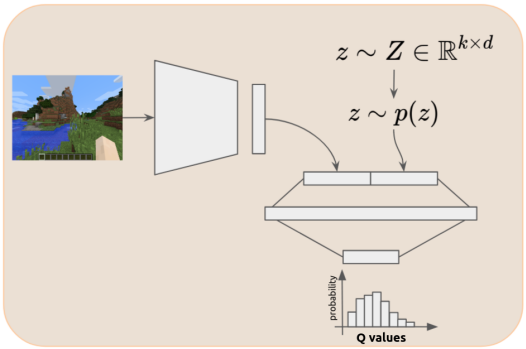


Figure 2. Skill-Learning pipeline for both contrastive and variational approaches.

In the last stage of the process, we aim to train a policy

$\pi(a|s, z)$ that maximizes the mutual information between inputs and the discovered latent variables. At each episode of the training, we sample a latent variable $z \sim p(z)$. Then, at each step, this embedding is concatenated with the embedding of the encoded observation at timestep t and forwarded through the network. This process is depicted in Figure 2. At this stage, the latent variables are interpreted as navigation goals. Hence, the agent is encouraged to visit those states that brings it closer to the navigation goal or latent variable. Since each of the latent variables encode different parts of the state distribution, this results in covering different regions for each z. Our method adopts the reverse form of the mutual information. Since we fix $p(z)$ as a uniform distribution we can remove the $\log p(z)$ constant term from the Equation 3, which results in the reward of Equation 7 for the variational case, and Equation 8 for the contrastive case. The agent receives a reward of 1 only if the embedding that is conditioning the policy is the closest to the encoded observation.

$$r(s, z) = q_\phi(z|s) \tag{6}$$

$$q_\phi(z = k|s) = \begin{cases} 1, & \text{if} \quad k = argmin_j ||z_e(s) - e_j|| \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

Where $z_e(s)$ is the encoded observation and e is the codebook of embeddings.

$$q_\phi(z = k|s) = \begin{cases} 1, & \text{for } k = argmax_j (z_e(s) W e_j) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

In the contrastive case, we compute a similarity measure. Therefore, we look for the latent index with highest similarity to the current observation.

5. Experiments

In this section we asses our method in two different Minecraft maps and set-ups. In each of the trainings we follow the pipeline described in Section 4, where we first generate random trajectories in the specified map. Later, we discover some categorical latents from them, and finally we train some conditioned policies on these latents. We refer the reader to the Appendix A to check the hyperparameters used in the different methods.

5.1. Discovery and learning of skills in handcrafted maps

The first map, in Figure 3a, is a 2D handcrafted map that consists in 9 different regions where the only difference among them is the floor type. The $p(s)$ induced from the random trajectories seen in Figure 3b covers the state subspace uniformly.

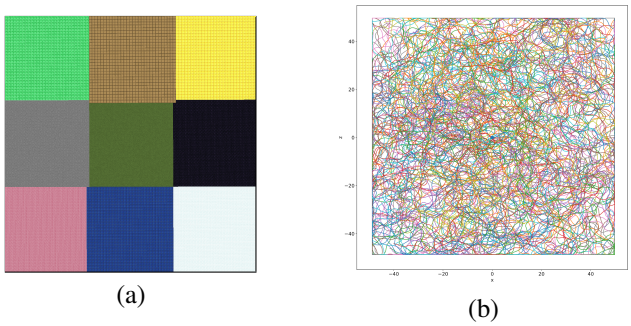


Figure 3. **(a)** Top-view of the toy map used for asses our method. The dimensions of this map are 100 square meters, if we consider that the Minecraft coordinate system was measured in meters. The agent has no exploratory issues even with a random policy. **(b)** Trajectories generated by a random agent deployed in any region of the map.

The *index maps* shown in Figure 4 are generated from the random trajectories in Figure 3b. We encode each observation from each trajectory with the index of the closest embedding in the latent codebook or K-Means centroids. We assign a color to each of the indexes and we get what we refer as an *index map*. Thanks to these maps, it is very easy to visualize whether the learned embeddings are meaningful or not.

Figure 4 compares the index maps learned with the variational and constrastive methods. In general we can see that the variational model discretizes slightly better, however it is unable to distinguish between two particular regions. Instead, despite the mixing and overlapping among different discovered latents, the contrastive approach manages to clusterize these regions decently. When studying these visualizations we should be aware that we are not taking into account the direction in which the agent was looking at the time it was recorded. Therefore, two points that are next to each other, might be encoding totally different views and hence will result in different latent codes.

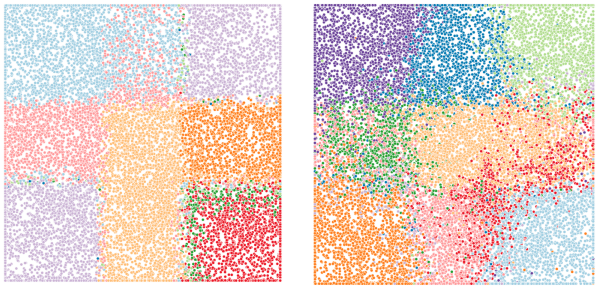
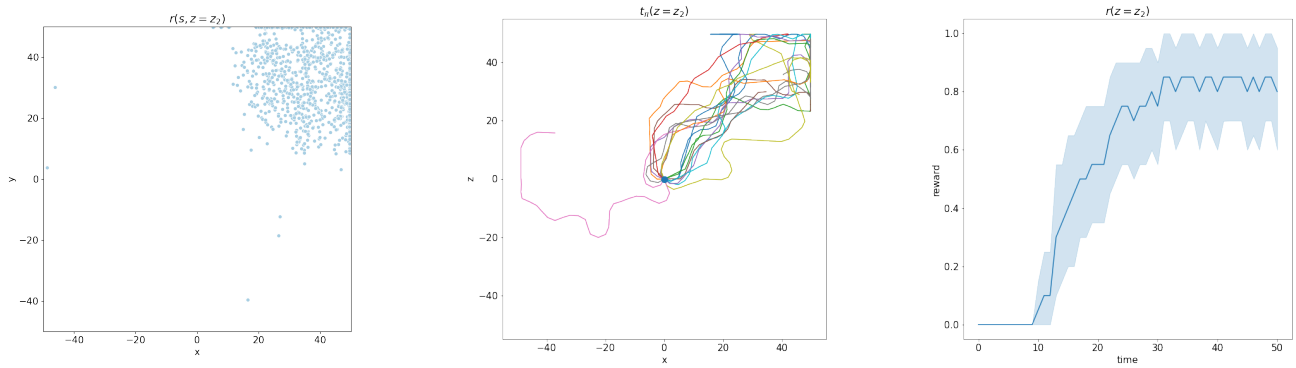


Figure 4. **Left:** Index map generated by the variational approach. **Right:** Index map generated by the contrastive approach.



(a) (b) (c)

Figure 5. **(a)** Observations that are encoded as the latent vector 2. **(b)** shows the trajectories performed by the agent and **(c)** the average reward received in those trajectories. Both plots conditioning the agent on the latent vector 2.

Learning representations from these observations is quite straightforward, since the dynamics are not very complex. Despite that, either when training with the contrastive and variational methods, we usually find latent centroids that encode observations in between plots. For this reason, if we wanted to have a categorical latent vector for each of the regions we could overclusterize the latent embeddings (in the contrastive case) or select a larger codebook (in the variational case). Since we train the agent in a map where we have already collected observations from random policies, we can plot the reward that those observations would be given when conditioned in each of the categorical latents. For instance, in Figure 5a we can see that the observations recorded in the top-right corner are encoded as the second discovered latent. Therefore, if we condition the policy with it, the agent will receive a reward of 1, following Equation 6.

In Figure 5b, we show how the performance of the agent when conditioned in this skill. All the trajectories head to the navigation goal except for one. Moreover, regardless of the initial direction, we can see that the agent is capable of orientating and go to the right direction. Lastly, in Figure 5c, we can see how it requires 10 steps to start triggering the intrinsic reward until reaching a stable 0.8 reward in average. For more examples on other latent codes we refer the reader to the Appendix B.

5.2. Discovery and learning of skills in realistic maps

The approach presented in the simple and handcrafted map of Section 5.1 fails when tested in more realistic Minecraft maps, like the one in Figure 6a. In this case, the discovered skills depicted in Figure 6b are no longer distributed in separable clusters over the map, and one skill (in light blue) tends to dominate over the others. Since we are treating each latent variable as a navigation goal, the ones that are spread out over the whole map will not guide but confuse the agent as it is receiving high reward in multiple places.

We overcome this problem by exploiting the spatial information provided by the environment in the form of coordinates. The learned embedding space contains a relationship between visual features and relative spatial coordinates with respect to the initial state, always the same one. This allows our agent to distinguish between two visually identical mountains located at two different positions in the map. Figure 6 shows complementary skills discovered from pixels (Figure 6b) or coordinates (Figure 6c). We would like to discover skills that take into account not only the visual similarity but also the position relative to the initial state, so we adopt a solution that considers the two types of state representations (Figure 6d).

We scale the VQ-VAE (van den Oord et al., 2017) to encode both pixels and coordinates independently. We sum up their embeddings and quantize the result to one of the codebook embeddings e. Now, the decoder of the model contains two independent branches, where one reconstructs an image and the other reconstructs a coordinate from the same embedding. The only difference from the original loss function is that we add a term for learning the coordinate reconstruction.

Once we have better latent embeddings for realistic maps, we can train an agent conditioned on them. Qualitative results are presented in Figure 7, where we can see an example of a latent encoding the bottom-left observations and the trajectories and rewards received in 10 evaluation episodes. The performance of all the conditioned policies is included in the Appendix C. We observed that at least 6 skills out of 10 could be potentially leveraged in a hierarchical policy.

Figure 8 shows a clear example of the pixels and coordinates mixing. If we based our encoder in just pixel observations we would have an almost identical latent encoding for both images. Instead, thanks to the coordinates, they become

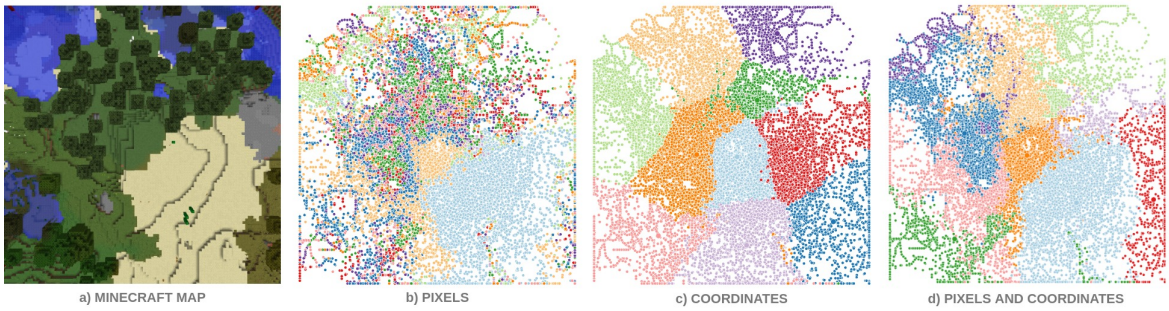


Figure 6. a) Top-view of our Minecraft map. The caption below b) c) d) refers to the nature of the states (pixels, coordinates or both) where all the trajectories were collected by random policies. Each coloured point indicates the closest centroid to the encoded embedding.

two separate latents. This may help the agent in being more precise and distinguish different regions of the space despite its visual similarity.

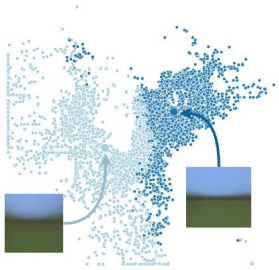
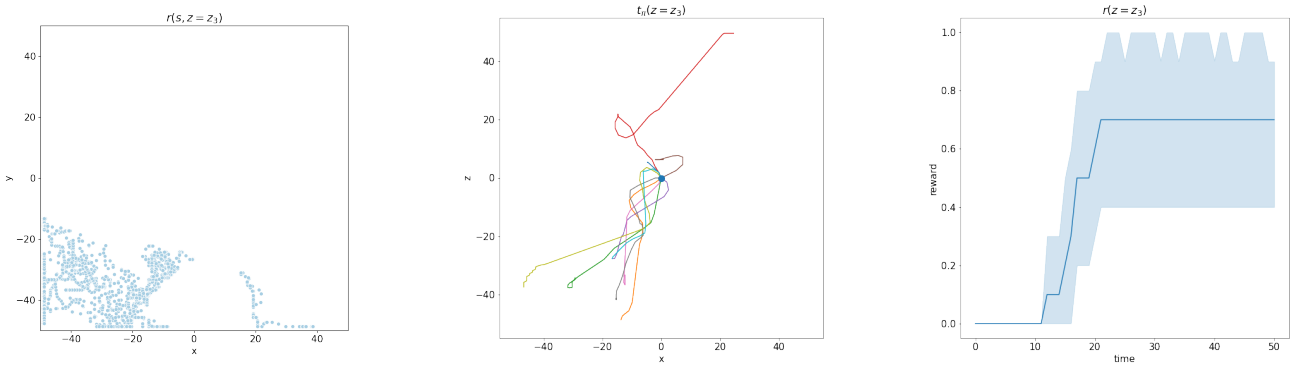


Figure 8. A very homogeneous and big region of the space becomes two different skills when using a combination of pixels and coordinates.

6. Conclusions

In this work we present a framework to extend the EDL paradigm to pixels. This requirement presents one main problem. Following the EDL derivation of the mutual information, we optimize the agents based on the MSE, which does not have any notion of space in the environment. To overcome that, we propose to derive the reverse form of the mutual information between inputs and latents. In this manner we compute the intrinsic reward in the embedding space which encodes the dynamics learned during the skill-discovery phase. We observed that contrastive and variational approaches are capable of learning the mapping from inputs to latents. In the former case, we leverage the use of positive pairs based on delayed observations, and the K-Means clustering algorithm for determining the categorical latents. In the latter, we make use of a VQ-VAE model that directly allows to infer categorical latents from the input data. We qualitatively show that some skills are correctly learned, specially in the toy maps where the environment dynamics are much easier. We demonstrate that for learning useful navigation goals in realistic maps we need more information, and therefore, we learn an embedding space combining pixel and coordinates observations.



a) b) c)

Figure 7. a) Observations that are encoded as the latent vector 3. b) and c) show the trajectories performed by the agent and the reward received in average when conditioned on skill 3.

References

J. Achiam, H. Edwards, D. Amodei, and P. Abbeel. Variational option discovery algorithms, 2018.

I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba. Hindsight experience replay. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5055–5065, 2017.

C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2020.

Y. Burda, H. Edwards, A. Storkey, and O. Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2018.

V. Campos, A. Trott, C. Xiong, R. Socher, X. Giro-i Nieto, and J. Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International Conference on Machine Learning*, pages 1317–1327. PMLR, 2020.

V. Campos, P. Sprechmann, S. Hansen, A. Barreto, S. Kapturowski, A. Vitvitskyi, A. P. Badia, and C. Blundell. Coverage as a principle for discovering transferable behavior in reinforcement learning. *arXiv preprint arXiv:2102.13515*, 2021.

M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

X. Chen, S. Xie, and K. He. An empirical study of training self-supervised visual transformers. *arXiv preprint arXiv:2104.02057*, 2021.

K. Cobbe, C. Hesse, J. Hilton, and J. Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pages 2048–2056. PMLR, 2020.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

C. Florensa, Y. Duan, and P. Abbeel. Stochastic neural networks for hierarchical reinforcement learning. In *International Conference on Learning Representations*, 2017.

D. Ghosh, A. Gupta, and S. Levine. Learning actionable representations with goal conditioned policies. In *International Conference on Learning Representations*, 2018.

K. Gregor, D. J. Rezende, and D. Wierstra. Variational intrinsic control. In *International Conference on Learning Representations (workshop)*, 2017.

W. H. Guss, M. Y. Castro, S. Devlin, B. Houghton, N. S. Kuno, C. Loomis, S. Milani, S. Mohanty, K. Nakata, R. Salakhutdinov, et al. The minerl 2020 competition on sample efficient reinforcement learning using human priors. *arXiv preprint arXiv:2101.11071*, 2021.

J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter. Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, 2(4):2096–2103, 2017.

D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2014.

M. Laskin, A. Srinivas, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pages 5639–5650. PMLR, 2020.

A. Lee, A. Nagabandi, P. Abbeel, and S. Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *Advances in Neural Information Processing Systems*, 33, 2020.

S. Mohamed and D. J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pages 2125–2133, 2015.

A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017.

C. Salge, C. Glackin, and D. Polani. Empowerment–an introduction. In *Guided Self-Organization: Inception*, pages 67–114. Springer, 2014.

T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal value function approximators.

J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

A. Stooke, K. Lee, P. Abbeel, and M. Laskin. Decoupling representation learning from reinforcement learning. *arXiv preprint arXiv:2009.08319*, 2020.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

A. Trott, S. Zheng, C. Xiong, and R. Socher. Keeping your distance: Solving sparse reward tasks using self-balancing shaped rewards. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2019.

A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6309–6318, 2017.

O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354, 2019.

D. Warde-Farley, T. Van de Wiele, T. Kulkarni, C. Ionescu, S. Hansen, and V. Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018.

D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Reinforcement learning with prototypical representations. *arXiv preprint arXiv:2102.11271*, 2021.

A. Hyperparameters

VQ-VAE	
learning rate	1e-3
batch size	256
num hiddens	64
num residual hiddens	32
num residual layers	2
embedding dim	256
num embeddings	10
β	0.25
decay	0.99

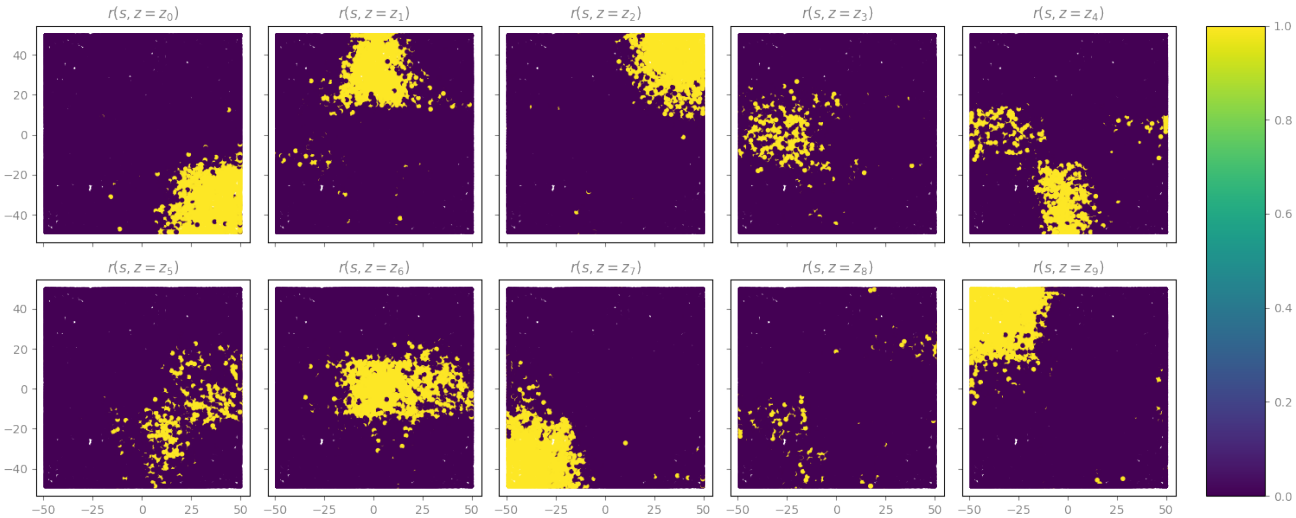
Contrastive	
learning rate	1e-3
batch size	128
τ	5e-3
soft update	2
embedding dim	128
k_μ	15
k_σ	5

Rainbow	
learning rate	2.5e-4
batch size	64
sampling	uniform
max episode steps	500
update interval	4
frame skip	10
gamma	0.99
clip delta	yes
num step return	1
adam eps	1.e-8
gray scale	no
frame stack	no
final expl frames	3.5e5
final epsilon	0.01
eval epsilon	0.001
replay capacity	10.e6
replay start size	1.e4
prioritized	yes
target upd interval	2.e4

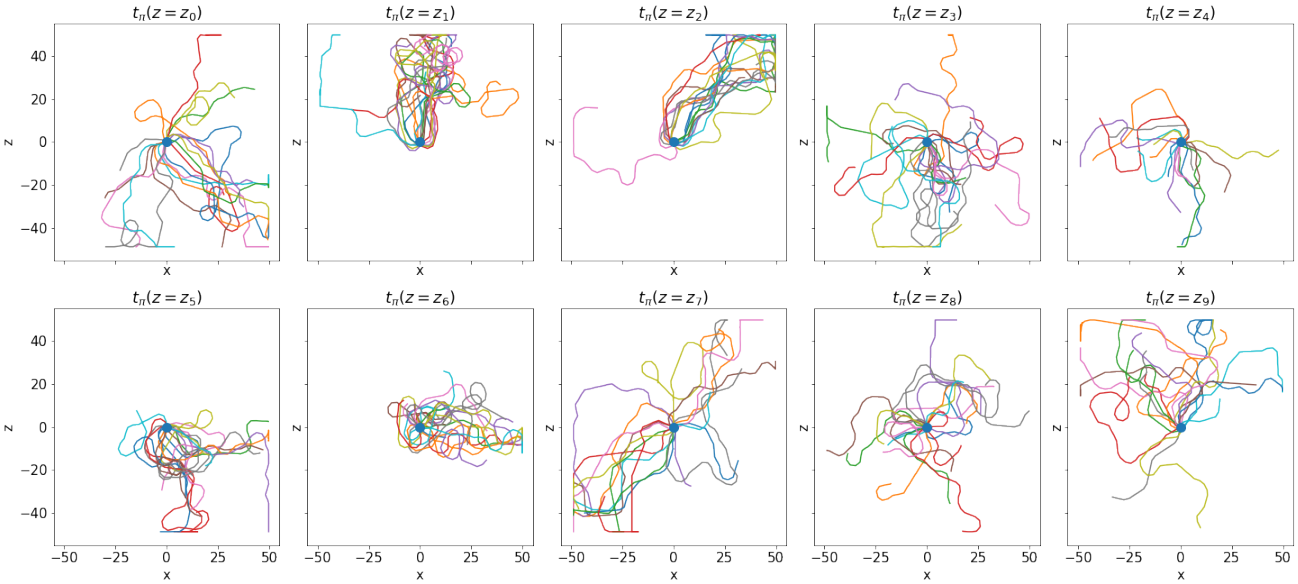
B. Learning from Pixels with Contrastive approach

(Handcrafted map)

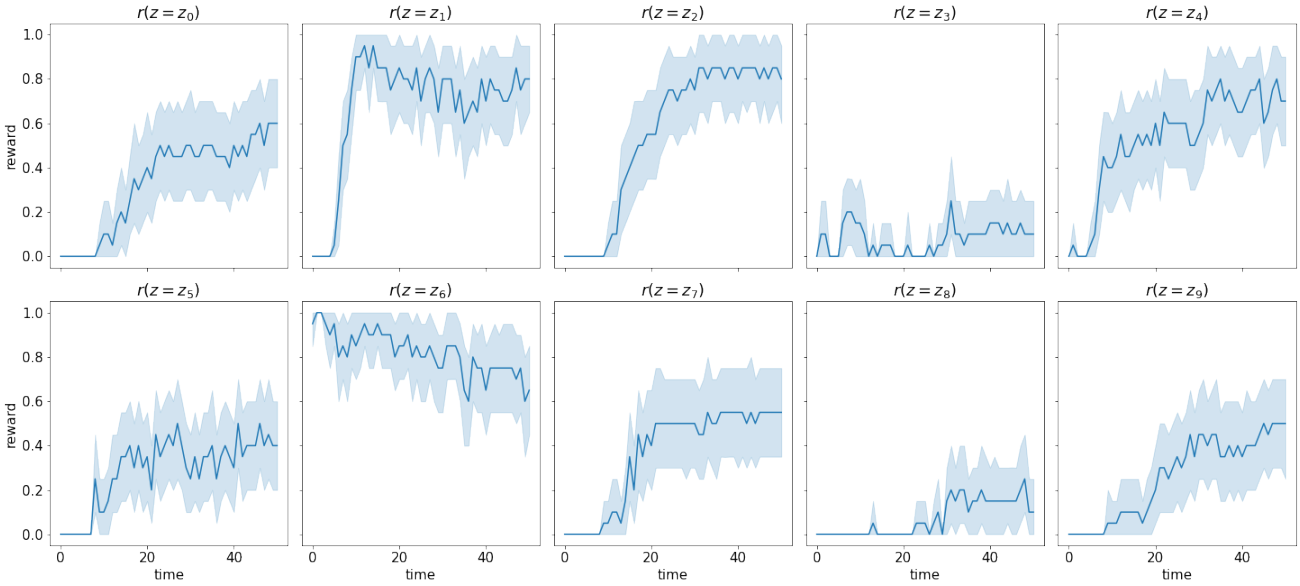
In this first plot, we show which observations would give a positive reward for each conditioning latent. In some cases they encode a very specific region, however, in some others the reward is spread over the map leading to poor policies.



As we can see in the evaluated trajectories, latents 0,1,2,4,5,6 are properly learned. Instead, we can see that latents 3,7,8,9 are more ambiguous and the agent does not solve the self-supervised task correctly.



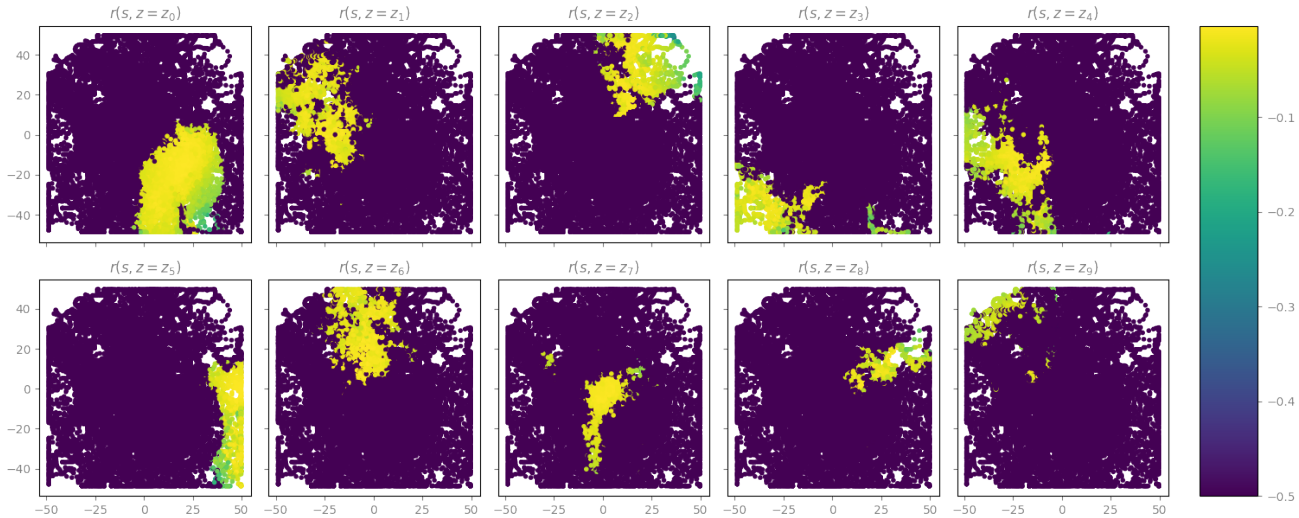
In the last plot we show the average reward of all the trajectories for each of the conditioning latents. We can see that the curves with higher reward match those trajectories in the previous plot that lead to the region encoded by the latent.


$r(z = z_0)$
$r(z = z_1)$
$r(z = z_2)$
$r(z = z_3)$
$r(z = z_4)$
$r(z = z_5)$
$r(z = z_6)$
$r(z = z_7)$
$r(z = z_8)$
$r(z = z_9)$
reward
time
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40

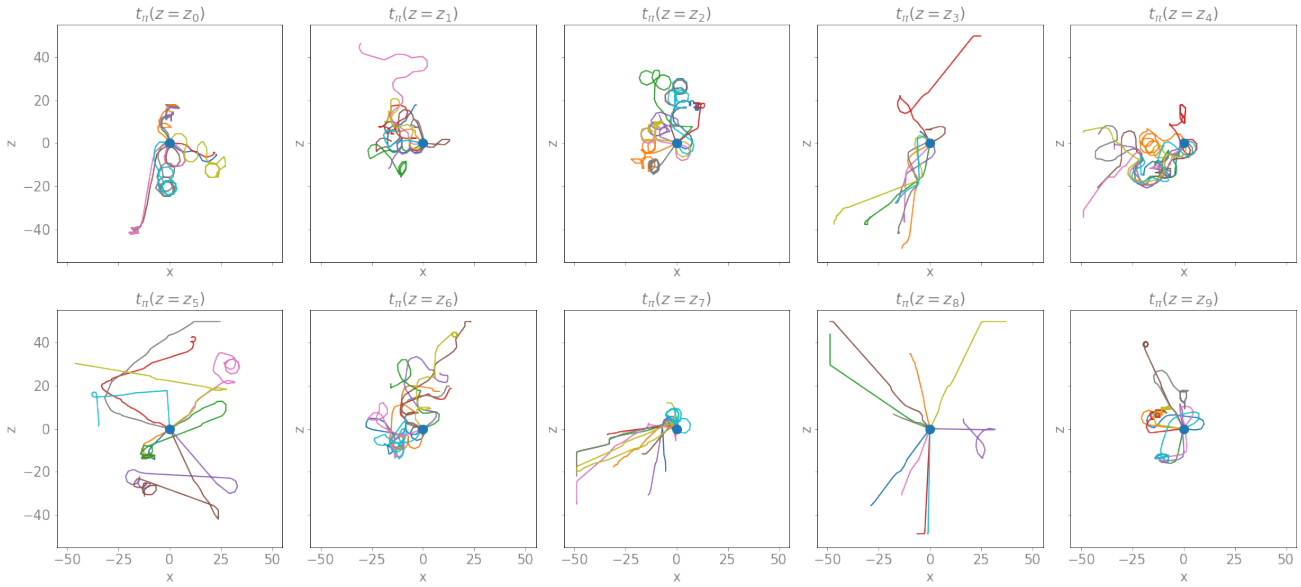
C. Learning from Pixels and Coordinates with Variational approach

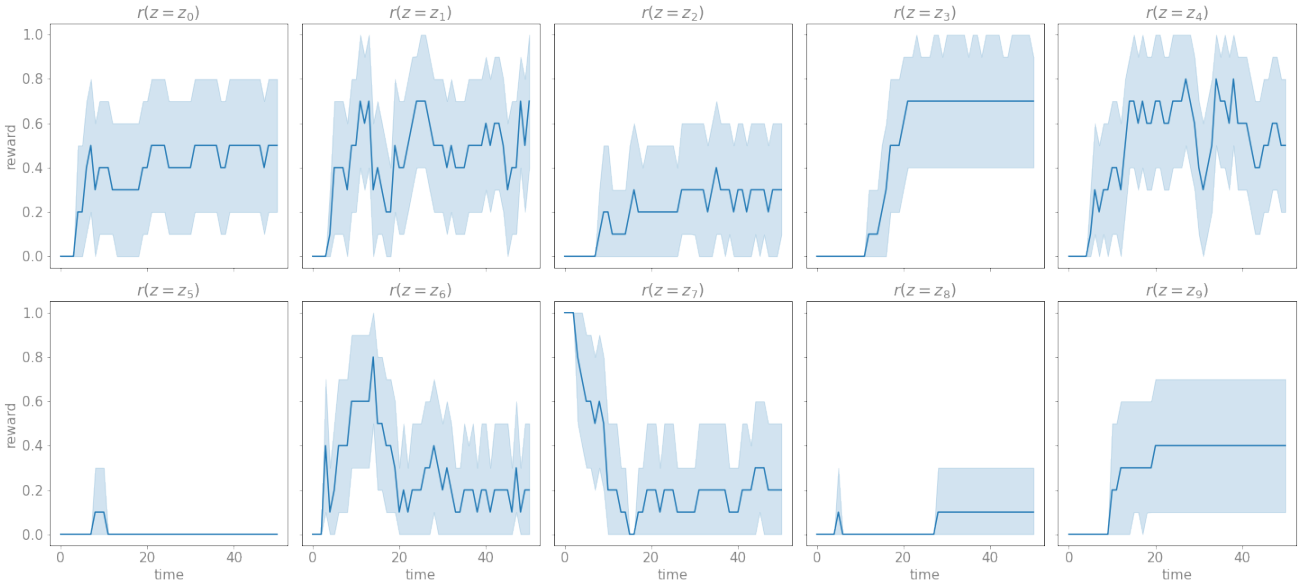
(Realistic map)

In this case, we can see that despite deploying the agent in a realistic map, the discovered latents encode very specific regions of the map. This is achieved by leveraging the raw pixel information and the relative position of the agent respect to the initial point.



Due to the dynamics of this map, we can see that is much more difficult to learn the right policies to guide the agent towards the regions encoded by the latents. Skills 1, 2, 3, 4, 6 and 7 still achieve good results, where, regardless of the agent's initial direction, it manages to reach the desired region.




$r(z = z_0)$
$r(z = z_1)$
$r(z = z_2)$
$r(z = z_3)$
$r(z = z_4)$
$r(z = z_5)$
$r(z = z_6)$
$r(z = z_7)$
$r(z = z_8)$
$r(z = z_9)$
reward
time
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40