DisTop: Discovering a Topological representation to learn diverse and rewarding skills

Arthur Aubret, Laetitia Matignon & Salima Hassas
Univ Lyon, Université Lyon 1, CNRS, LIRIS F-69622, Villeurbanne, France
{firstname.lastname}@univ-lyon1.fr

***Abstract*—An efficient way for a deep reinforcement learning agent to explore in sparse-rewards settings can be to learn a set of skills that achieves a uniform distribution of terminal states. We introduce DisTop, a new model that simultaneously learns diverse skills and focuses on improving rewarding skills. DisTop progressively builds a discrete topology of the environment using an unsupervised contrastive loss, a growing network and a goal-conditioned policy. Using this topology, a state-independent hierarchical policy can select which skill to execute and learn. In turn, the new set of visited states allows an improved learnt representation. Our experiments emphasize that DisTop is agnostic to the ground state representation and that the agent can discover the topology of its environment whether the states are high-dimensional binary data, images, or proprioceptive inputs. We demonstrate that this paradigm is competitive on MuJoCo benchmarks with state-of-the-art algorithms on both single-task dense rewards and diverse skill discovery without rewards. By combining these two aspects, we show that DisTop outperforms a state-of-the-art hierarchical reinforcement learning algorithm when rewards are sparse. We believe DisTop opens new perspectives by showing that bottom-up skill discovery combined with dynamic-aware representation learning can tackle different complex state spaces and reward settings.**

***Index Terms*—Intrinsic motivation, Deep reinforcement learning, hierarchical learning, developmental learning.**

I. Introduction

In reinforcement learning (RL), an autonomous agent learns to solve a task by interacting with its environment [1] thus receiving a reward that has to be hand-engineered by an expert to efficiently guide the agent. However, when the reward is sparse or not available, standard methods badly explore their environment [2]. One way to improve exploration is to provide to the agent prediction-based [3] or count-based [4] flat rewards, making the agent reach surprising or unvisited states. These bonuses are called *intrinsic rewards* because the agent self-generates its reward. However, such methods only learn one policy that sequentially targets new areas; this sequential exploration prevents the simultaneous exploration of several new areas, making the agent forget how to reach them, thereby hurting exploration [5].

This issue partially motivated methods where an agent hierarchically commits to a temporally extended behavior named skills or *options* [6]. For example, such skills can be targeting a position in a maze environment or manipulating torques to make a humanoid walk. While it is possible to learn hierarchical skills with an extrinsic (*i.e.* task-specific) reward [7], it does not fully address the exploration issue. Another possibility, which we follow here, is to take inspiration from how infants learn, which is called developmental learning. It has been argued that infants autonomously learn hierarchically organized skills thanks to a spontaneous exploration named intrinsic motivation [8], [9]. To simulate such learning process, one can incite an agent to acquire a large and diverse repertoire of skills learned thanks to intrinsic rewards. In this learning schema, the agent samples goals (often states) and the corresponding skill tries to reach the goal. While the methods that follow this paradigm manage to learn a large set of different skills that simultaneously explore different areas, they 1- are learned during a pre-training phase, preventing online skill specialization to a task [10]; 2- require an expertly built space of subjectively interesting features for defining skills, *e.g.* (x, y) coordinates of an agent [11].

We are interested in scaling skill-learning methods to all reward settings and state spaces, typical of how humans learn in different settings using different modalities/inputs (images, proprioceptive, chess board, . . .). This would make it possible to use the same algorithm on different reward settings (sparse, dense, no-rewards), independently of the ground state space.

In this paper, we introduce DisTop, a hierarchical and developmental algorithm that simultaneously learns to explore the state space, using an intrinsic motivation to visit rarely seen states, and to solve a task by maximizing an extrinsic reward. DisTop relies on several components. Skills are goal-conditioned low-level policies that aim at reaching a particular goal state chosen in the state space. To be agnostic to the kind of state space (image, high-dimensional proprioceptive elements, etc. . .), DisTop builds a continuous representation (called an embedding) of the state space based on a InfoNCE-like loss [12]. This embedding is learned to preserve a particular topology of the state space, where the Euclidian distance between two close states captures the minimal number of actions required to get from one state to the other. A dynamical clustering of the embedded space allows a tractable estimation of the current probability density of visited embedded states and of the value, in terms of reward, of states.

Thus, during a learning episode, a high-level policy can choose a goal state (or high-level action) that is both valuable and rarely seen, using a skewed density of probability on the embedded states. This goal state is considered as a skill (low-level goal-conditioned policy) which is executed to generate a new state-action-state-reward trajectory. The trajectory is used to 1) refine the current skill, 2) improve the embedding, thus leading to an adaptation of the clustering and of the estimations of state visited probabilities and values.

Our experimental contribution shows the genericity and efficiency of selecting skills to improve using a topological representation. This contribution is 4-folds: 1- we visualize the representation learned by DisTop and exhibit that DisTop is agnostic to the shape of the ground state space and works with states being images, high-dimensional proprioceptive data or high-dimensional binary data; 2- we show that DisTop competes with a state-of-the-art (SOTA) algorithm specific to single-task dense rewards settings thanks to its ability to select rewarding skills; 3- we demonstrate that DisTop competes with a SOTA algorithm on diverse skills learning benchmarks, with relative performance depending on how well the learned embedding is adequate for the considered environment; 4- importantly, because it selects goals that maximize both the density of visited states and the extrinsic rewards, it outperforms several hierarchical baselines in sparse-reward environments.

II. Background

In this section, we explain the necessary background to understand DisTop. In section II-A, we describe the framework of goal-conditioned RL. Then, since DisTop uses a representation learning method that builds on top of InfoNCE, we explain the InfoNCE objective in section II-B. Finally, since DisTop skews a given probability density distribution similarly to Skew-Fit [13], we describe the framework of Skew-Fit [13] in section II-C.

A. *Goal-conditioned reinforcement learning for skill discovery*

In episodic RL, the problem is modelled with a Markov Decision Process (MDP). An agent interacts with an unknown environment in the following way: the agent gets a state $s_0 \in S$ that follows an initial state distribution $\rho_0(s)$. Its policy π selects actions $a \in A$ to execute depending on its current state $s \in S$, $a \sim \pi(\cdot|s)$. Then, a new state is returned according to the transition dynamics $s' \sim p(\cdot|s,a)$. The agent repeats the procedure until it reaches a particular state or exceeds a fixed number of steps T. This loop is called an episode. An agent learns π to maximize the expected cumulative discounted reward $\mathbb{E}_{\pi}\big[\sum_{t=0}^{T}\gamma^t R^{ext}(s_{t-1},a_{t-1},s_t)\big]$ where γ is the discount factor and R^{ext} the extrinsic reward function. The policy π_θ can be approximated with a neural network parameterized by θ [14]. In our case, the agent tries to learn *skills*, defined as a policy that targets a goal-state (or *goal* g_t). This goal is provided by a high-level policy. At each timestep, we define an interaction as a tuple $(s_{t-1},a_{t-1},s_t,g_t,R^{ext}(s_{t-1},a_{t-1},s_t))$. To learn skills, the agent computes an intrinsic reward according to the chosen skill [2]. Following *Universal Value Function Approximators* [15], the intrinsic reward and the skill become dependent on the chosen goal g_t: $r_t^g = R_\omega(s_{t-1},a_{t-1},s_t,g_t)^{int}$ and $\pi_\theta^g(\cdot|s_t) = \pi_\theta(\cdot|s_t,g_t)$. With this scheme, we can relabel an interaction with an arbitrary goal and recompute the intrinsic reward [16].

B. *Contrastive learning and InfoNCE*

Contrastive learning aims at discovering a similarity metric between data points [17]. First, positive pairs that represent similar data and fake negative pairs are built. Second, they are given to an algorithm that computes a data representation able to discriminate the positive from the negative ones. InfoNCE [18] proposed to build positive pairs from states that belong to the same trajectory, making the process scale to all input representations. The negative pairs are built with states randomly sampled. For a sampled trajectory, it builds a set $\mathbb{B}$ that concatenates $N-1$ negative states and a positive one s_{t+p}. Then, it maximizes $\mathbb{L}^N_{InfoNCE} = \mathbb{E}_{\mathbb{B}}\big[\log \frac{f_p(s_{t+p},v_t)}{\sum_{s_j\in\mathbb{B}} f_p(s_j,v_t)}\big]$ where v_t is a representation of a sequence of previous states learned with an autoregressive model, p indicates the additional number of steps needed to select the positive sample and f_p are positive similarity functions. The numerator brings together states that are part of the same trajectory, and the denominator pushes away negative pairs.

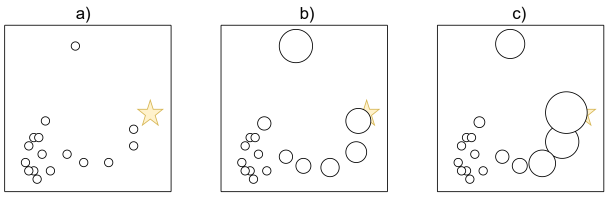

Fig. 1: a) States visited in the environment, where one circle corresponds to one visitation of state. b) **Skew-Fit** over-weights states situated in rarely visited areas. c) **DisTop** over-weights states according to both their density and their proximity with extrinsic rewards (the star).

C. *Skew-Fit*

Skew-Fit [13] strives to learn goal-conditioned policies that visit a maximal entropy of states, where all states are considered as possible goals. Assuming the agent visited a set of states as illustrated in Figure 1a), it learns a generative model that incrementally over-samples states situated in low-density areas (figure 1b). While learning and executing skills that target these state-goals, the density distribution of visited states progressively tend towards a uniform density distribution $U(s)$.

Formally, assuming it learns a ψ-parameterized generative model $q_\psi(s)$, *e.g.* with a Variational Auto-Encoder (VAE) [19], Skew-Fit uses importance sampling to skew the state density distribution closer to the uniform one. Thus, they maximize $\mathbb{E}_{p(s)} q_\psi(s)^{1+\alpha_{skew}} \log q_\psi(s)$ where $p(s)$ denotes the true actual probability density distribution of visited states and $\alpha_{skew} < 0$ defines the importance of low-density states. If $\alpha_{skew} = 0$, the density distribution will stay similar; if $\alpha_{skew} = -1$, it strongly over-samples low-density state-goals so that $p(s)$ quickly converge to $U(S)$. In practice, they show that directly sampling states with $s \sim \frac{1}{Z} q_\psi(s)^{1+\alpha_{skew}}$, where Z is the normalization constant, reduces the variance.

III. Method

a) Problem statement: we are interested in learning skills that simultaneously maximize the state entropy and extrinsic rewards when provided, thereby supposing the ability of a

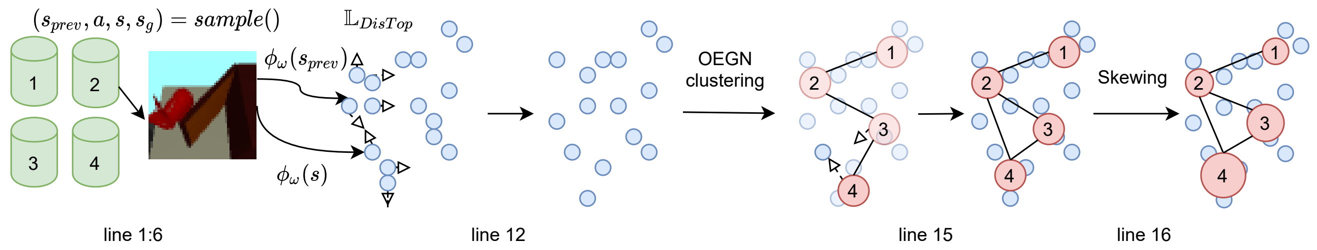

Fig. 2: Illustration of a learning step of our growing network and contrastive loss (cf. text). Cylinders are buffers associated to each cluster, blue circles are states embedded with ϕ and pink circles represent clusters. The image is an example of state in the *Visual Door* [20] environment. Lines indicate the corresponding line in algorithm 1.

high-level policy to select extrinsically rewarding/unvisited states as goals for a low-level policy.

b) Overview of DisTop: DisTop is a hierarchical algorithm that selects the skill (the goal of a goal-conditioned policy) to execute in the environment at the beginning of each episode. Figure 2 illustrates the **learning** loop of DisTop. To be agnostic to the state space, it learns embeddings of states $\phi_\omega(\cdot)$ that locally capture the distance in terms of number of actions between them. Two consecutive states become close in the embedding space, whatever are their ground state representations. We call such representation a *dynamics-aware* representation. To sample rarely visited or rewarding embedded states as goals, we need to approximate a skewed probability density distribution of embedded states. To do this in a tractable way, DisTop discretizes/clusters the embedded states with an algorithm named "Off-policy Embedded Growing Network" (OEGN) and associates interactions to clusters according to how the next embedded state of the interaction is clustered. DisTop estimates both the mean extrinsic reward achieved while targeting a goal-state of the cluster and the density of a visited state by counting the number of embedded states its clusters contain. Finally, DisTop selects extrinsically rewarding/unvisited clusters and randomly samples a state that belongs to it, such that the goal-conditioned policy has to reach that state.

A. Learning a dynamics-aware state representation

In this section, we explain how DisTop learn the *state representation* function ϕ_ω that maps a given state to an embedded space. Intuitively, DisTop will learn ϕ_ω so that the following entropy maximization will discover states distant from visited states in terms of actions.

To build such continuous representation, DisTop takes advantage of the InfoNCE loss (cf. section II-B). Similarly to [21], we do not consider the whole sequence of interactions, since this makes it more dependent on the policy. Typically, a constant and deterministic optimal policy would make the representations equally distant, since all pairs of states would be positive. DisTop considers its local variant and builds positive pairs with only two consecutive states, *i.e.* pairs of states that are separated by one action. This way, it keeps distinct the states that cannot be consecutive and it more accurately matches the topology of the environment. We select our similarity function as $f_\omega(s_t, s_{t+1}) = e^{-k||\phi_\omega(s_t)-\phi_\omega(s_{t+1})||_2}$ where ϕ_ω is a neural network parameterized by ω and k is a temperature hyperparameter [22]. If $\mathbb{B}$ is a batch of N different consecutive pairs, the local InfoNCE objective, LInfoNCE, is described by:

$$\mathbb{L}_{LInfoNCE} = \mathbb{E}_{(s_t,s_{t+1})\in\mathbb{B}} \log \frac{f_\omega(s_t, s_{t+1})}{\sum_{s\in\mathbb{B}} f_\omega(s_t, s)}$$
$$\geq \mathop{\mathbb{E}}_{s_t,s_{t+1}\in\mathbb{B}} - k||\phi_\omega(s_t) - \phi_\omega(s_{t+1})||_2 \quad (1)$$
$$- \log(1 + \sum_{s\in\mathbb{B}_{s\neq s_{t+1}}} f_\omega(s, s_{t+1})). \quad (2)$$

We introduce this lower bound since we found it to stabilize the learning process. Intuitively, equation 2 brings together consecutive states, and pushes away states that are separated by many actions. There are several drawbacks with this objective function. Firstly, the representation may be strongly **dilated**, *i.e.* the agent may cover too much distance with one action. In this case, too many clusters may be assigned to an area, preventing their practical use for density estimation. Secondly, the DRL algorithm requires semantically stable inputs to compute Q-values: if the representation of its goals quickly changes, it cannot take into account the changes and may output bad approximations of Q-values. We tackle these issues by reformulating the learning objective as two-folds constrained maximization (cf. equation **??** in appendix **??**) in order to keep our representation stable and correctly dilated. In practice, in order to perform a stochastic gradient descent, we transform the constraints into penalties and maximize:

$$\mathbb{L}_{DisTop} = \mathbb{E}_{(s_t,s_{t+1})\in\mathbb{B}}\big[- k_c(ReLU(||\phi_\omega(s_t) - \phi_\omega(s_{t+1})||_2^2 - \delta^2) - \log(1 + \sum_{s\in\mathbb{B}} f_\omega(s, s_{t+1})) - \beta||\phi_\omega(s_{t+1}) - \phi_{\omega'}(s_{t+1})||_2^2\big]. \quad (3)$$

Using equation 3, DisTop forces the distance between consecutive states to be below a threshold δ, avoiding the collapse of parts of the representation; $k_c \geq 1$ is the dilatation coefficient that brings together consecutive states. Secondly, β encourages our representation to stay consistent over time, weights ω' being a slow exponential moving average of ω. Thus, DisTop learns a consistent representation that keeps close consecutive states while avoiding collapse. In fact, by modifying k_c and k, one can select the level of dilatation and of the representation (cf. appendix **??** for an analysis). One can notice that the function depends on the density of consecutive states in $\mathbb{B}$; we discuss the density distribution of states that feeds $\mathbb{B}$ in section III-C.

B. Clustering the embeddings to estimate the probability density distribution of embedded states

We need to approximate the probability density distribution of embedded states to skew it to favor exploration and maximize extrinsic rewards. Since our representation strives to maintain a level of dilatation of the representation, DisTop can match the dynamics-aware topology of the environment by clustering the embedded states. This allows us to estimate the probability density distribution of embedded states by counting the number of embedded states that belong to a cluster. Using this approximation, the next section details how we can skew the sampling of cluster, before sampling goal-states, to favor exploration and maximize extrinsic rewards.

a) Clustering algorithm: In order to adapt the clustering to temporally-related and goal-related interactions, we propose a new variant of the Growing When Required (GWR) [23], which we call Off-policy Embedded Growing Network (OEGN). In comparison with a VAE, OEGN 1- has a smaller number of parameters; 2- also allows to skew according to extrinsic rewards (section III-C). OEGN dynamically creates, moves and deletes clusters so that clusters generate a network that covers the whole set of embedded states, independently of their density. Each node/cluster $c \in C$ has a reference vector w_c which represents its position in the input space. Using this reference vector, an embedding is assigned to its closest cluster, *i.e.* $min_c||\phi_{\omega'}(s) - w_c)||_2 \leq \delta_{new}$ where δ_{new} is the radius of the ball and the threshold for creating clusters. δ_{new} is particularly important since it is responsible for the granularity of the clustering: a low value will induce a lot of small clusters, while a high one induce few large clusters that badly approximate the topology (cf. appendix **??**). A learned topology can be visualized in Figure 6.

Essentially, the *delete* operator removes unused nodes, the *creation* operator creates a node if an embedding falls outside the radius of all clusters and the *moving* operator resembles the k-means clustering. Edges are added between two clusters when the agent passes from one cluster to the other, and deleted after a cluster-normalized delay of uselessness. Overall, these operators make sure that each embedding belongs to the fixed-radius ball centered on the center of a cluster. In comparison with the GWR, OEGN accounts for the topology link between two successive states (*creation* of nodes/edges) and prevents an excessive creation of clusters (*creation* of nodes) which can occur when non-optimized state representations make successive embedded states too distant. OEGN also speeds up the deletion of wrongly created clusters (node *delete* operator). Technical details about OEGN and GWR can be found in appendix **??**.

b) Approximating the probability density distribution of embedding goals and states: we assume that the visit density of a state region is reflected by the probability of a state to be in its cluster. Thus, we approximate the density of a state with the density parameterized by w, reference vector of $e = \phi_{\omega'}(s)$:

$$q_w(e) \approx \frac{count(c_e)}{\sum_{c' \in C} count(c')}$$

where $count(c_e)$ denotes the number of embedded states that belong to the cluster that contains e. While our approximation $q_w(s)$ decreases the quality of our density approximation, it makes our algorithm efficient: we associate a buffer to each cluster and only compute the density of clusters. In practice, we trade off the bias of $q_w(s)$ and the instability of OEGN by decreasing δ_{new}: the smaller the clusters are, the smaller the bias is, but the more unstable is OEGN (cf. appendix **??**).

Using this approximation, we just need to keep states in the buffer of their closest node to sample from an arbitrary density distribution. In practice, we associate to each node a second buffer that takes in interactions according to the proximity of the original goal with respect to the node. This results in two sets of buffers, $\mathbb{B}^G$ and $\mathbb{B}^S$, to respectively sample goal-states and states according to the desired density distribution (*e.g.* a uniform one). Formally, to sample a given interaction (s_{prev}, a, s_g, s), DisTop can select the cluster $\arg\min_{c \in C} ||\phi(s) - c||_2$ in $\mathbb{B}^S$ or the cluster $\arg\min_{c \in C} ||\phi(s_g) - c||_2$ in $\mathbb{B}^G$.

C. Selecting novel or rewarding skills

While sampling low-density or rewarding states is attractive to solve a task, it is not easy to sample new reachable goals. For instance, using an embedding space $\mathbb{R}^3$, the learned embedded topology of the environment may only exploit a small part of it, such that randomly sampling goals, that are not direct embedding of states, inside $\mathbb{R}^3$ would mostly result in unreachable goals. Typically, one can see in figure 5 that embeddings of states only exploit a small part of the whole space. To make sure goals are reachable, DisTop generates goals by sampling and embedding previously visited states (similarly to [24]).

To sample the goals-states or states, DisTop samples a cluster with a skew distribution and a state. This sampling procedure applies for both selecting a goal to execute and sampling a mini-batch, but with different desired probability density distribution of embedded states. We will now explain how DisTop skews the actual probability density distribution of embedded states.

a) Skewing the sampling of goal-states: similarly to Skew-Fit and in order to sample new exploratory goals for the goal-conditioned policy, DisTop increases the entropy of visited terminal states by skewing the current density distribution of visited states (cf. section II-A) with

$$\begin{aligned} p_{\alpha_{skew}}(c) &= \frac{count(c)^{1+\alpha_{skew}}}{\sum_{c' \in C} count(c')^{1+\alpha_{skew}}} \\ &= \frac{e^{(1+\alpha_{skew}) \log count(c)}}{\sum_{c' \in C} e^{(1+\alpha_{skew}) \log count(c')}}, \end{aligned} \tag{4}$$

where α_{skew} is the skewed parameter (cf. section II). It is equivalent to sampling with a high-level Boltzmann policy π^h, using a novelty bonus reward $\log count(c)$ and a fixed temperature parameter $1+\alpha_{skew}$. We can add a second reward to modify our skewed high-level policy π^h to take into consideration extrinsic rewards. We associate to a cluster $c \in C$ a value R^{ext}_c that represents the average extrinsic rewards received by the skills associated to its goals:

$$R^{ext}_c = \mathbb{E}_{\substack{s\in c,\, g=\phi_\omega(s),\, a_t\sim\pi^g_\theta(\cdot|s_t),\\ s_{t+1}\sim p(\cdot|s_t,a_t)}} R^{ext}(s_t,a_t,s_{t+1}) \quad (5)$$

where the extrinsic value of a cluster R^{ext}_c is updated with an exponential moving average To favor exploration, we can also update the extrinsic value of the neighbors of the final state's cluster with a slower learning rate (cf. appendix **??**). Our sampling rule can then be:

$$\pi^h(c) = \underset{C}{\text{softmax}}\big(\underbrace{t^{ext}R^{ext}_c +}_{\text{Rewarding cluster}} \underbrace{(1+\alpha_{skew})\log count(c)}_{\text{Low-density cluster}}\big)$$

where t^{ext} is a weighting temperature hyperparameter. Finally, the high-level policy selects a cluster $c_1 \sim \pi^h(\cdot)$, its state-buffer $\mathbb{B}^S_{c_1}$ and samples a state with an approximation of the uniform density distribution inside the cluster's c_1 ball of radius δ_{new} (cf. appendix **??**).

b) Skewing the sampling of a learning minibatch: our goal-conditioned policy π^g_θ needs a uniform density distribution of goals to avoid forgetting previously learned skills, while our representation function ϕ_ω requires a uniform density distribution over visited states to quickly learn the representation of novel areas. Therefore, to construct a minibatch, DisTop samples clusters $\sim p_{\beta_{skew}}$ with $\beta_{skew} \neq \alpha_{skew}$, randomly takes half of buffers in $\mathbb{B}^G$ and the rest from $\mathbb{B}^S$ and samples interactions. In practice, we keep $1+\beta_{skew}=0$ such that DisTop uniformly samples clusters to learn from, thereby not prioritizing the skill learning process according to the visitation count. DisTop can also sample a *Ratio* of clusters with π^h if one does not care about forgetting skills (cf. section IV).

D. Summary algorithm

Algorithm 1 sums up the algorithm of DisTop. First (line 1:6), the agent samples interactions $i=(s_{prev},a,s,s_g)$ from previously built buffers, where s_g is the original goal-state. Second (line 12), DisTop learns to embed the states s with a neural network $e=\phi_\omega(s)$ and a contrastive loss such that the embeddings reflect the inherent topology of the environment. Third (line 15), OEGN creates, deletes and moves clusters to track the representations of states. Finally (line 16), rather than skewing the untractable probability density distribution of states, we can easily skew the probability distribution of clusters. The skewing is done by computing the mean reward achieved while targeting a goal-state of the cluster and the density of a visited state by counting the number of embedded states its clusters contain.

During execution (cf. algorithm 2) and at the beginning of each episode, a Boltzmann policy π_h selects a cluster c' according to the skewed distribution illustrated in figure 1c) (line 1); then DisTop selects a state s that belongs to the buffer B^S_c of the selected cluster to be a goal state $s=s_g$ and computes its embedding $g=\phi(s_g)$ (line 2). A goal-conditioned policy

Algorithm 1: Learning iteration of DisTop. The batch size BS and $Ratio$ are hyperparameters

```
 1:                                ▷ Skewed sampling of batch (section III-C)
 2: Sample $BS * Ratio$ clusters $c_1 \sim \pi^h(\cdot)$.
 3: Sample $BS*(1-Ratio)$ clusters $c_2 \sim p_{\beta_{skew}}(\cdot)$
 4: Sample buffers: $\mathbb{B}_1 \sim U(\{\mathbb{B}^G_{c_1}, \mathbb{B}^S_{c_1}\})$
 5:                 $\mathbb{B}_2 \sim U(\{\mathbb{B}^G_{c_2}, \mathbb{B}^S_{c_2}\})$
 6: Build interactions batch $i = cat(i_1 \sim \mathbb{B}_1, i_2 \sim \mathbb{B}_2)$.
 7:                                                    ▷ Prepare updates
 8: Compute embeddings of goals $g=\phi_{\omega'}(s_g)$
 9: Compute embeddings of observations $e=\phi_{\omega'}(s)$
10: Relabel goals (cf. appendix ??).
11:                                                     ▷ Update models
12: Update representations $\phi_\omega$ (section III-A).
13: Update $\omega'$, exponential moving average of $\omega$
14: Update $\pi^g_\theta$ with $r=||e-g||_2$ and SAC [14]
15: Update OEGN (section III-B and appendix ??)
16: Update $\pi^h$ and $p_{\alpha_{skew}}$ according to cluster assignments.
```

Algorithm 2: Execution part of DisTop for each episode.

```
1: Sample $c' \sim \pi^h(\cdot)$ and $s_g \sim \mathbb{B}^G_{c'}$
2: Compute embedding of goal $g=\phi_{\omega'}(s_g)$
3: Execute $\pi^g$ to gather interactions $i=(s_{prev},a,s,s_g)$
4: Store each $i$ in $\mathbb{B}^G_{c'}$ and $\mathbb{B}^S_{\arg\min_{c\in C}||\phi_{\omega'}(s)-c||_2}$
```

π^g_θ, trained to reach g, acts in the environment and discovers new reachable states close to its goal (line 3). The interactions $i=(s_{prev},a,s,s_g)$ made by the policy are stored in a buffer according to their initial objective (B^G_c where c clusters s_g) and the embedding of the reached state (B^S_c where c clusters s) (line 4).

IV. Experiments

Our experiments aim at studying whether the ability of DisTop to select low-density/rewarding goals in a dynamics-aware latent space makes the approach generic to different task settings and state spaces. We compare DisTop to SOTA algorithms (Skew-Fit, ELSIM, LESSON) on three kinds of environments with different ground state spaces. All curves are a smooth averaged over 5 seeds, except for ablation experiments which are averaged over 3 seeds, completed with a shaded area indicating the mean +/- standard deviation over seeds. We provide used hyperparameters in appendix **??**, detail environments and evaluation protocols in appendix **??**, images of achieved goals in appendix **??**.

A. Does DisTop discover the environment topology even though the ground state space is unstructured ?

In this experiment, we analyze the representation learned by a VAE and DisTopaccording to the ground state space. We make sure that the learned representation is visualizable. Therefore, we adapted the *gym-minigrid* environment [25] (cf. Figure 3) where a randomly initialized agent moves for fifty

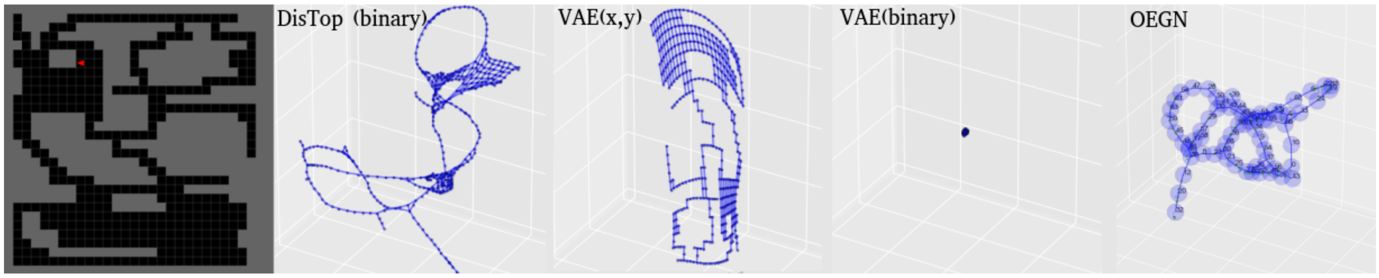

Fig. 3: Visualization of the representations learned by a VAE and DisTop on the environment displayed at the far left. From left to right, we respectively see a- the rendering of the maze; b- the continuous representation learned by DisTop with 900-dimensional binary inputs; c- a VAE representation with true (x, y) coordinates; d- a VAE representation with 900-dimensional binary inputs; e- OEGN network learned from binary inputs.

timesteps in the four cardinal directions. Each observation is a 900-dimensional one-hot vector with 1 at the position of the agent (*binary*) like in section IV-A. Interactions are given to the representation learning algorithm. In Figure 3, we clearly see that DisTop is able to discover the topology of its environment since each of the connected points are distinct but close with each other. In contrast, when accessing the 900-dimensional binary representation of states, the VAE representation collapses since it cannot take advantage of the proximity of states in the ground state space; it only learns a correct representation when it gets (x, y) positions, which are well-structured. This toy visualization highlights that, unlike VAE-based models, **DisTop does not depend on well-structured ground state spaces to learn a suitable environment representation to evaluate the density of a state.**

B. Is DisTop able to learn diverse skills without rewards ?

We assess the diversity of learned skills on two robotic tasks, *Visual Pusher* (a robotic arm moves in 2D and eventually pushes a puck) and *Visual Door* (a robotic arm moves in 3D and can open a door) [20]. These environments are particularly challenging since the agent has to learn skill from 48x48 images using continuous actions without accessing extrinsic rewards. We compare to the SOTA algorithm Skew-Fit [10], which skews the visited state density distribution in the ground state space with a VAE [19] and periodically updates its representation. We use the same evaluation protocol than Skew-Fit: a set of diverse images are manually pre-generated to be diverse (the same set is used for each evaluation step), given to the representation function and the goal-conditioned policy executes the skills that target each embedding of images. The score of a skill is computed in different ways according to the environment: in the *Visual Pusher* environment, we compare the final distance of the position puck with its desired position; in the *Visual Door* environment, we compare the angle of the door with the desired angle. We average the score of all skills. We do not compare DisTop to LESSON, ELSIM and SAC since they were designed only to solve a task and can not learn without rewards.

In Figure 4(left), we observe that skills of DisTop are learned quicker on *Visual Pusher*, but are slightly worse on *Visual Door*. Since both Skew-Fit and DisTop generates rewards with the L2 distance, we hypothesize that this is due to the structure of the learned goal space. In practice, we observed that DisTop is more stochastic than Skew-Fit, probably because the intrinsic reward function is required to be smooth over consecutive states. It results that a one-step complete change of the door angle creates a small change in the representation, thereby a small negative intrinsic reward. Despite the stochasticity, it is able to reach the goal, as evidenced by the minimal distance reached through the episode (Distop(min)). Stochasticity does not bother the evaluation on *Visual Pusher* since the arm moves elsewhere after pushing the puck in the right position. Therefore, **DisTop manages to learn slightly more or less diverse skills than Skew-fit, depending on the environment structure.**

C. Can DisTop solve non-hierarchical dense rewards tasks?

We test DisTop on MuJoCo environments *Halfcheetah-v2* and *Ant-v2* [26], where the agent gets rewards to move forward as fast as possible. We fix the maximal number of timesteps to 300. We compare DisTop to SAC [14] and ELSIM [27], a method that follows the same paradigm as DisTop (see section VI). We do not compare to Skew-Fit since it was designed only to learn without extrinsic rewards and it can not leverage extrinsic rewards. We obtain better results than in the original ELSIM paper using similar hyperparameters to DisTop. In Figure 4 (Right), we observe that DisTop obtains high extrinsic rewards and clearly outperforms ELSIM. It also outperforms SAC on *Halfcheetah-v2* at 1.5 million steps and is close to SAC on *Ant-v2*. In contrast, we highlight that SAC overfits to dense rewards settings and cannot learn in sparse or no-reward settings (see below). **Despite the genericity of DisTop and the specificity of SAC to dense rewards settings (see below), DisTop competes with SAC on two environments.**

D. Does combining representation learning, entropy of states maximization and task-learning improve exploration on high-dimensional hierarchical sparse-rewards tasks ?

We evaluate DisTop ability to explore and optimize rewards on image-based versions of *Ant Maze* and *Point Maze* environments [11]. The state is composed of a proprioceptive state of the agent and a top view of the maze ("Image"). Details

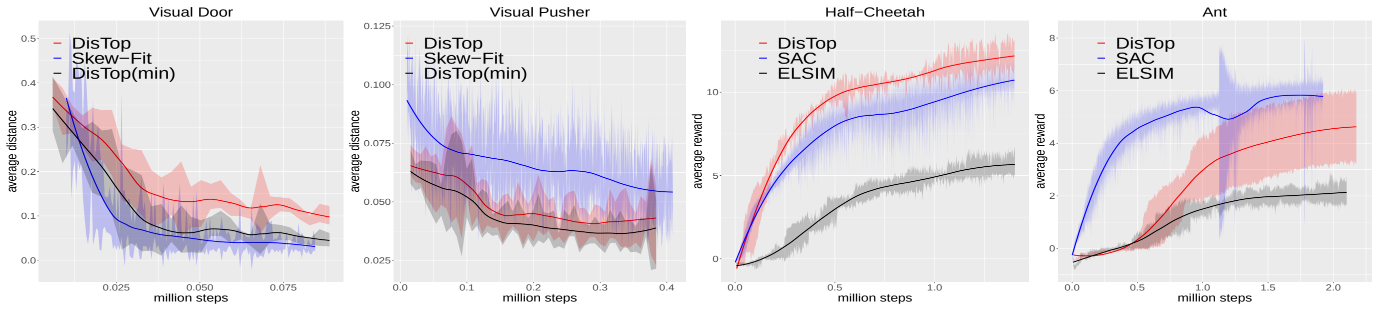

Fig. 4: **Left:** Comparison of DisTop and Skew-Fit on their ability to reach diverse states. The smaller the average distance the better, see text for information about how we compute the distance in each environment. DisTop(min) is the minimal distance reached through an evaluation episode. At each evaluation iteration, the distances are averaged over fifty goals. **Right:** Average rewards gathered throughout episodes of 300 steps while training on *Halfcheetah-v2* and *Ant-v2* environments.

about state and action spaces are given in appendix **??**. In contrast with previous methods [21], we remove the implicit *curriculum* that changes the extrinsic goal across episodes; here we train only on the farthest goal and use a sparse reward function. Thus, the agent gets a non-negative reward only when it gets close to the top-left goal. We compare our method with a SOTA hierarchical method, LESSON [21] since it performs well on hierarchical environments like mazes, ELSIM, SAC [14], a prediction-based flat method mixing LWM [28] with DisTop (LWM-DisTop), a DisTop's variant that rewards a single policy with a count-based bonus like [29] (FlatDisTop). Similarly to section IV-C), we do not compare to Skew-Fit since it was not designed to solve a task. We only pass the "image" to the representation learning part of the algorithms, assuming that an agent can separate its proprioceptive state from the sensorial state. In figure 5, we can see that DisTop is the only method that manages to regularly reach the goal; in fact, looking at a learned 3D OEGN network in figure 6, we can see that it successfully represents the U-shape form of the maze. LWM-DisTop too quickly learns to predict, making the bonus useless for exploration. FlatDisTop outperforms DisTop, suggesting that DisTop's ability to retain previous skills can slow down its skill discovery. LESSON discovers the goal but does not learn to return to it[1]; we hypothesize that this is because, unlike DisTop, it does maximize the entropy of states, and thus hardly reach the goal. Neither SAC nor ELSIM find the reward. We suppose that the undirected width expansion of the tree of ELSIM does not maximize the state-entropy, making it spend too much time in useless areas and thus inefficient for exploration. **Therefore, DisTop outperforms two hierarchical methods in two sparse rewards environment by prioritizing its goal sampling process on low-entropy areas.**

V. Ablation Study

In this section, we analyze and visualize the role and the components of our model: hyperparameters of equation 3, our dynamical clustering and the goal sampling process.

[1]In *Point Maze*, the best seed of LESSON returns to the goal after 5 millions timesteps

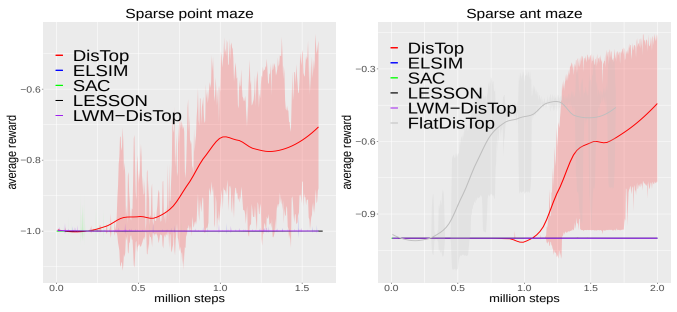

Fig. 5: Average rewards gathered throughout training episodes.

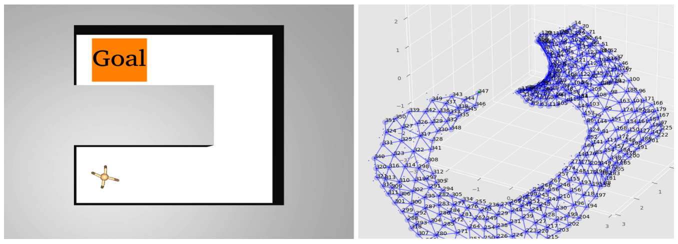

Fig. 6: Top view of Ant Maze environment with its goal position and an example of OEGN network learned by DisTop.

A. Does DisTop control the dilatation of the representation ?

We want to assess if equation 3 enables to control the dilatation of the representation learned by DisTop. To quantitatively analyze the dilatation of the representation, we compute the mean distance between two consecutive embedded states; we learn a representation in the gridworld since actions are discrete, randomly chosen, and we removed not-moving transitions. Using a discrete action space avoids biasing the metric by the distribution of actions output by the policy. We observe in figure 8 that a low dilatation coefficient indeed results in a large dilatation of the representation. We further qualitatively analyze hyperparameters of equation 3 in appendix **??**.

B. What are the important components of DisTop?

We aim at understanding the importance of each novel component of DisTop. In figure 7a), we compare the performance of DisTop according to different levels of dilatation

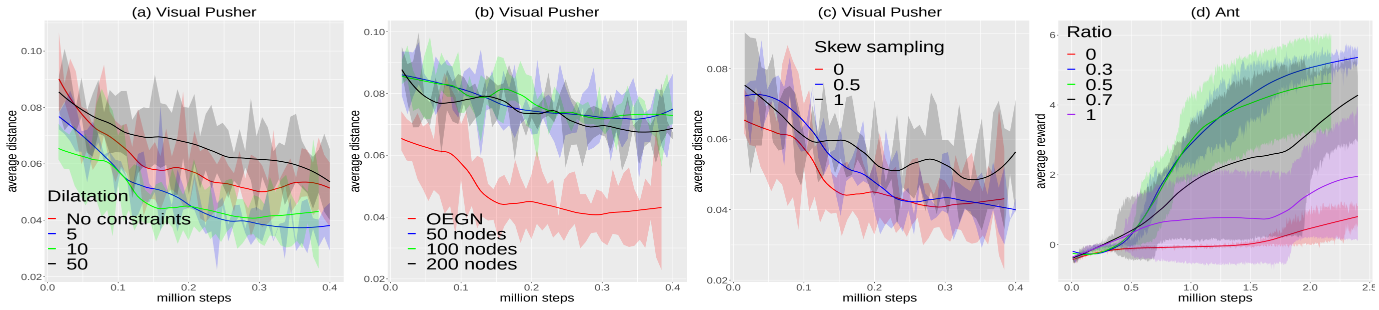

Fig. 7: Same environments and evaluation protocols than in figure 4. The agent aims at minimizing the evaluated distance and increasing the average reward. a- comparison of performance in *Visual Pusher* between using no constraints in equation 3 and several values of k_c; b- comparison in *Visual Pusher* of DisTop with OEGN and using a fix number of nodes as in a SOM; c- performance of DisTop in *Visual Pusher* for several skew sampling coefficients $1+\alpha_{skew}$; d- evaluation of DisTop in *Ant* for several values of $ratio$.

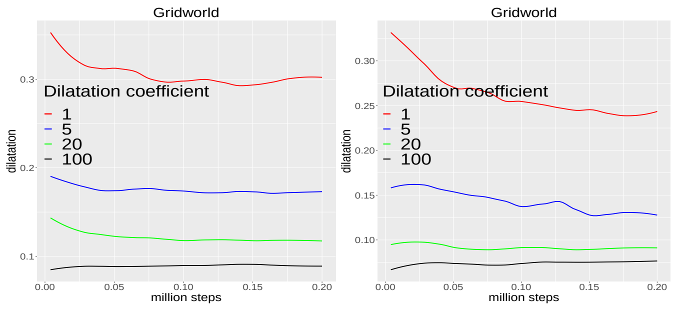

Fig. 8: Dilatation of the representation according to the dilatation parameter k_c with squaring (left) and without squaring (right).

with the same cluster creation threshold δ_{new}. We see that fixing a k_c compatible with the size of a cluster determines the quality of the density approximation, and thus the novelty of the selected goals (see also section **??**); using only the consistency penalty of equation 3 (*No constraints*), the agent learns slower than with constrains, probably because it underestimates the density of states in some areas. Figure 7b shows the importance of dynamically creating clusters by comparing OEGN with a Self-Organizing Map (SOM) [30] that learns with a fix number of nodes. We observed that our OEGN approximately creates 100 clusters on average on *Visual Pusher*, so we tried 50, 100 and 200 nodes. We clearly see that fix numbers of nodes struggle to generate new skills. We hypothesize that this is due to the resulting bad approximation of the state density: at the beginning, all clusters have a low density while the agent visited few different states. Figure 7c stresses out the importance of skewing the goal-state sampling process. The agent focuses its goal-state selection process on low-density areas with $1+\alpha_{skew}$ to speed up representation and policy learning. Overall, DisTop is robust to small changes of $1+\alpha_{skew}$. Finally, figure 7d shows the importance of sampling a $ratio$ of learning interactions using π^h in dense rewards settings (here *Ant*). It results that the agent purposely avoids learning on badly rewarding areas (the policy and representation) so that it does not create clusters in this area and reserves its learning batch for interactions of rewarding skills. At the extremes, trying to learn all skills stuck the agent in local optima because it tries to retain too many skills ($Ratio = 0$), while learning only in rewarding areas prevents exploration ($Ratio = [1, 0.7]$). To sum up, **our penalties and OEGN are critical to well-approximate a density distribution of visited states and skewing it is crucial to efficiently select the skills to learn on.**

VI. Related works

a) Intrinsic skills in hierarchical RL: some works propose to learn hierarchical skills, but do not introduce a default behavior that maximizes the visited state entropy, limiting the ability of an agent to explore. For example, it is possible to learn skills that target a ground state or a change in the ground state space [11], [31]. These approaches do not generalize well with high-dimensional states. Therefore, one may want to generate rewards with a learned representation of goals, but current methods still rely on hierarchical random walks [21], [32] or learn the representation during pre-training [33]. We have shown in section IV that DisTop explores quicker by maximizing the entropy of states in its topological representation. DCO [34] generates *options* by approximating the second eigenfunction of the combinatorial graph Laplacian of the MDP. It extends previous works [35], [36] to continuous state spaces. Independently of our work, HESS [37] maximizes a count-based bonus in a dynamics-aware embedding space, thereby managing to solve the sparse-rewards mazes. But they use a fixed partition of the whole embedding to compute the bonus, making it unclear how they could use a latent embedding with more than two or three dimensions.

b) Intrinsic motivation to learn diverse skills: DisTop simultaneously learns skills, their goal representation, and which skill to train on. It contrasts with several methods that focus on selecting *which skill to train on* assuming a good goal representation is available [38]–[40]. They either select goals according to a curriculum defined with intermediate difficulty, the learning progress (LP) [41] or by imagining new language-based goals [42]. In addition, DisTop strives to learn either skills that are diverse or extrinsically rewarding. It differs from a set of prior methods that learn only diverse skills during a pre-training phase, preventing exploration for end-to-end learning. Previous methods [43], [44] learn a discrete set of skills by maximizing the mutual information (MI) between states and skills, but they hardly generalize over skills. It has been extended to continuous sets of skills, using a generative model [45] or successor features [46]. In both case, directly maximizing this MI may incite the agent to focus only on simple skills [47]. DISCERN [24] maximizes the MI between a skill and the last state of an episode using a contrastive loss and the true goal to generate positive pairs. Several methods take advantages of a VAE to learn skills, but our method better scales to any ground state space (see appendix **??**). Typically, RIG [20] uses a VAE to compute a goal representation before training the goal-conditioned policies. Using a VAE, one can define a frontier with a reachability network, from which the agent should start stochastic exploration [48]; but the gradual extension of the frontier is not automatically discovered, unlike approaches that maximize the entropy of states (including DisTop). Skew-Fit [13] learns more diverse skills by making the VAE over-weight low-density states. It is unclear how Skew-Fit could target another density distribution over visited states than a uniform one. Approaches based on LP have already been built over VAEs [49], [50]; we believe that DisTop could make use of LP to improve skill selection.

c) Skill discovery for end-to-end exploration: like DisTop, ELSIM [27] discovers diverse and rewarding skills in an end-to-end way. It builds a tree of skills and selects the branch to improve with extrinsic rewards. DisTop outperforms ELSIM for both dense and sparse-rewards settings (cf. section IV). This end-to-end setting has also been experimented through multi-goal distribution matching [51], [52] where the agent tries to reduce the difference between the density of visited states and a given density distribution (with high-density in rewarding areas). Yet, either they approximate a state density distribution over the ground state space [52] or assume a well-structured state representation [51]. Similar well-structured goal space is assumed when an agent maximizes the reward-weighted entropy of goals [53].

d) Dynamics-aware representations: a set of RL methods try to learn a topological map without addressing the problem of discovering new and rewarding skills. Some methods [54]–[56] consider a topological map over direct observations, but to give flat intrinsic rewards or make planning possible. Other approaches learn landmarks for planning, but need pre-training with uniform initial state and goal density distributions [57], [58] or need a uniform initial state density [59], thereby avoiding the exploration challenge. We emphasize that SFA-GWR-HRL [60] hierarchically takes advantage of a topological map built with two GWR placed over two Slow Feature Analysis algorithms [61]; it is unclear whether it can be applied to other environments than their robotic setting. Functional dynamics-aware representations can be discovered by making the distance between two states match the expected difference of trajectories to go to the two states [62]; interestingly, they exhibit the interest of topological representations for HRL and propose to use a fix number of clusters to create goals. Previous work also showed that an active dynamics-aware search of independent factors can disentangle the controllable aspects of an environment [63]. Other methods take advantage of temporal contrastive losses for other functional uses; therefore, unlike DisTop, they do not try to learn a topology of the environment while controlling the dilatation of the representation. For example, successor representations [34], [64] orthogonalize the features of the representation and some methods use a temporally-contrastive representation for imitation [65], end-to-end task solving [66], [67] or flat exploration [68], [69]. [57] also cluster a dynamics-aware embedding, but they learn a fixed number of size-changing clusters, making them inappropriate for estimating the inherent density distribution of states.

VII. Conclusion

We introduced a new developmental and hierarchical learning model, DisTop, that simultaneously learns a discrete topology of its environment and the skills that navigate into it. This topology allows to efficiently select the skills to improve regardless of the ground state space or reward setting, *i.e* extrinsically rewarding ones or exploratory ones. This way, DisTop competes with baselines from dense rewards setting thanks to its ability to prioritize the learning of extrinsically rewarding goals. Furthermore, DisTop competes with a baseline from the no rewards setting on one over two environments, presumably because it can be more efficient to maximize the entropy of the probability density distribution of embedded states rather than the one of states. Finally, it outperforms several hierarchical baselines in the sparse rewards setting because DisTop can both explore and maximize extrinsic rewards in an end-to-end way. There are exciting perspectives. HRL and planning based approaches [70] could both take advantage of the topology and make easier states discovery. More importantly, disentangling the topology [71] could improve the scalability of the approach since each combination of (pusher, puck) positions in *Visual Pusher* induces a new state that must be visited. Learning the topology of each independent factors of variations, like pusher and puck positions, may better support generalization over these factors of variations.

VIII. Acknowledgements

We thank Alain Dutech for his comments and suggestions on the manuscript. We gratefully acknowledge support from GENCI–IDRIS (Grant 2021-AD011011623) and the CNRS/IN2P3 Computing Center (Lyon - France) for providing computing and data-processing resources needed for this work. This work has been partially funded by ANR project DeLiCio (ANR-19-CE23-0006).

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[2] A. Aubret, L. Matignon, and S. Hassas, "A survey on intrinsic motivation in reinforcement learning," *arXiv preprint arXiv:1908.06976*, 2019.

[3] Y. Burda, H. Edwards, D. Pathak, A. J. Storkey, T. Darrell, and A. A. Efros, "Large-scale study of curiosity-driven learning," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[4] M. G. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, Eds., 2016, pp. 1471–1479.

[5] A. Ecoffet, J. Huizinga, J. Lehman, K. O. Stanley, and J. Clune, "First return, then explore," *Nature*, vol. 590, no. 7847, pp. 580–586, 2021.

[6] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[7] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[8] F. Guerin, "Learning like a baby: a survey of artificial intelligence approaches," *The Knowledge Engineering Review*, vol. 26, no. 2, pp. 209–236, 2011.

[9] G. Baldassarre, T. Stafford, M. Mirolli, P. Redgrave, R. M. Ryan, and A. Barto, "Intrinsic motivations and open-ended development in animals, humans, and robots: an overview," *Frontiers in psychology*, vol. 5, p. 985, 2014.

[10] V. Pong, M. Dalal, S. Lin, A. Nair, S. Bahl, and S. Levine, "Skew-fit: State-covering self-supervised reinforcement learning," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020, pp. 7783–7792.

[11] O. Nachum, S. S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., 2018, pp. 3303–3313.

[12] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018.

[13] V. Pong, M. Dalal, S. Lin, A. Nair, S. Bahl, and S. Levine, "Skew-fit: State-covering self-supervised reinforcement learning," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020, pp. 7783–7792.

[14] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1861–1870.

[15] T. Schaul, D. Horgan, K. Gregor, and D. Silver, "Universal value function approximators," in *International conference on machine learning*. PMLR, 2015, pp. 1312–1320.

[16] M. Andrychowicz, D. Crow, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *Annual Conference on Neural Information Processing Systems*, 2017, pp. 5048–5058.

[17] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[18] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018.

[19] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014.

[20] A. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., 2018, pp. 9209–9220.

[21] S. Li, L. Zheng, J. Wang, and C. Zhang, "Learning subgoal representations with slow dynamics," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=wxRwhSdORKG

[22] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[23] S. Marsland, J. Shapiro, and U. Nehmzow, "A self-organising network that grows when required," *Neural Networks*, vol. 15, no. 8-9, pp. 1041–1058, 2002.

[24] D. Warde-Farley, T. V. de Wiele, T. D. Kulkarni, C. Ionescu, S. Hansen, and V. Mnih, "Unsupervised control through non-parametric discriminative rewards," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[25] M. Chevalier-Boisvert and L. Willems, "Minimalistic gridworld environment for openai gym," https://github.com/maximecb/gym-minigrid, 2018.

[26] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *IEEE/RSJ IROS*. IEEE, 2012, pp. 5026–5033.

[27] A. Aubret, L. Matignon, and S. Hassas, "ELSIM: end-to-end learning of reusable skills through intrinsic motivation," in *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Proceedings, Part II*, ser. Lecture Notes in Computer Science, F. Hutter, K. Kersting, J. Lijffijt, and I. Valera, Eds., vol. 12458. Springer, 2020, pp. 541–556.

[28] A. Ermolov and N. Sebe, "Latent world models for intrinsically motivated exploration," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[29] A. L. Strehl and M. L. Littman, "An analysis of model-based interval estimation for markov decision processes," *Journal of Computer and System Sciences*, vol. 74, no. 8, pp. 1309–1331, 2008.

[30] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[31] A. Levy, R. Platt, and K. Saenko, "Hierarchical reinforcement learning with hindsight," in *International Conference on Learning Representations*, 2019.

[32] O. Nachum, S. Gu, H. Lee, and S. Levine, "Near-optimal representation learning for hierarchical reinforcement learning," *arXiv preprint arXiv:1810.01257*, 2018.

[33] X. Lu, S. Tiomkin, and P. Abbeel, "Predictive coding for boosting deep reinforcement learning with sparse rewards," *CoRR*, vol. abs/1912.13414, 2019.

[34] Y. Jinnai, J. W. Park, M. C. Machado, and G. Konidaris, "Exploration in reinforcement learning with deep covering options," in *International Conference on Learning Representations*, 2019.

[35] M. C. Machado, M. G. Bellemare, and M. H. Bowling, "A laplacian framework for option discovery in reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 2017, pp. 2295–2304.

[36] A. Bar, R. Talmon, and R. Meir, "Option discovery in the absence of rewards with manifold analysis," in *International Conference on Machine Learning*. PMLR, 2020, pp. 664–674.

[37] S. Li, J. Zhang, J. Wang, Y. Yu, and C. Zhang, "Active hierarchical exploration with stable subgoal representation learning," *arXiv preprint arXiv:2105.14750*, 2021.

[38] C. Florensa, D. Held, X. Geng, and P. Abbeel, "Automatic goal generation for reinforcement learning agents," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, ser. Proceedings of Machine Learning Research, J. G. Dy and A. Krause, Eds., vol. 80. PMLR, 2018, pp. 1514–1523.

[39] C. Colas, P. Fournier, O. Sigaud, and P. Oudeyer, "CURIOUS: intrinsically motivated multi-task, multi-goal reinforcement learning," *CoRR*, vol. abs/1810.06284, 2018.

[40] Y. Zhang, P. Abbeel, and L. Pinto, "Automatic curriculum learning through value disagreement," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[41] P.-Y. Oudeyer and F. Kaplan, "What is intrinsic motivation? a typology of computational approaches," *Frontiers in neurorobotics*, vol. 1, p. 6, 2009.

[42] C. Colas, T. Karch, N. Lair, J. Dussoux, C. Moulin-Frier, P. F. Dominey, and P. Oudeyer, "Language as a cognitive tool to imagine goals in curiosity driven exploration," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[43] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, "Diversity is all you need: Learning skills without a reward function," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[44] C. Florensa, Y. Duan, and P. Abbeel, "Stochastic neural networks for hierarchical reinforcement learning," in *5th International Conference*

on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net, 2017.

[45] A. Sharma, S. Gu, S. Levine, V. Kumar, and K. Hausman, "Dynamics-aware unsupervised discovery of skills," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[46] S. Hansen, W. Dabney, A. Barreto, D. Warde-Farley, T. V. de Wiele, and V. Mnih, "Fast task inference with variational intrinsic successor features," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[47] 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*. PMLR, 2020, pp. 1317–1327.

[48] H. Bharadhwaj, A. Garg, and F. Shkurti, "Leaf: Latent exploration along the frontier," *arXiv e-prints*, pp. arXiv–2005, 2020.

[49] G. Kovač, A. Laversanne-Finot, and P.-Y. Oudeyer, "Grimgep: learning progress for robust goal sampling in visual deep reinforcement learning," *arXiv preprint arXiv:2008.04388*, 2020.

[50] A. Laversanne-Finot, A. Pere, and P.-Y. Oudeyer, "Curiosity driven exploration of learned disentangled goal spaces," in *Conference on Robot Learning*. PMLR, 2018, pp. 487–504.

[51] S. Pitis, H. Chan, S. Zhao, B. Stadie, and J. Ba, "Maximum entropy gain exploration for long horizon multi-goal reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7750–7761.

[52] L. Lee, B. Eysenbach, E. Parisotto, E. P. Xing, S. Levine, and R. Salakhutdinov, "Efficient exploration via state marginal matching," *CoRR*, vol. abs/1906.05274, 2019.

[53] R. Zhao, X. Sun, and V. Tresp, "Maximum entropy-regularized multi-goal reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7553–7562.

[54] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-parametric topological memory for navigation," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=SygwwGbRW

[55] N. Savinov, A. Raichuk, D. Vincent, R. Marinier, M. Pollefeys, T. P. Lillicrap, and S. Gelly, "Episodic curiosity through reachability," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[56] B. Eysenbach, R. Salakhutdinov, and S. Levine, "Search on the replay buffer: Bridging planning and reinforcement learning," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 15 220–15 231.

[57] L. Zhang, G. Yang, and B. C. Stadie, "World model as a graph: Learning latent landmarks for planning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 611–12 620.

[58] Z. Huang, F. Liu, and H. Su, "Mapping state space using landmarks for universal goal reaching," *Advances in Neural Information Processing Systems*, vol. 32, pp. 1942–1952, 2019.

[59] M. Laskin, S. Emmons, A. Jain, T. Kurutach, P. Abbeel, and D. Pathak, "Sparse graphical memory for robust planning," 2020.

[60] X. Zhou, T. Bai, Y. Gao, and Y. Han, "Vision-based robot navigation through combining unsupervised learning and hierarchical reinforcement learning," *Sensors*, vol. 19, no. 7, p. 1576, 2019.

[61] L. Wiskott and T. J. Sejnowski, "Slow feature analysis: Unsupervised learning of invariances," *Neural computation*, vol. 14, no. 4, pp. 715–770, 2002.

[62] D. Ghosh, A. Gupta, and S. Levine, "Learning actionable representations with goal conditioned policies," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=Hye9lnCct7

[63] E. Bengio, V. Thomas, J. Pineau, D. Precup, and Y. Bengio, "Independently controllable features," *CoRR*, vol. abs/1703.07718, 2017.

[64] Y. Wu, G. Tucker, and O. Nachum, "The laplacian in rl: Learning representations with efficient approximations," in *International Conference on Learning Representations*, 2018.

[65] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain, "Time-contrastive networks: Self-supervised learning from video," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 1134–1141.

[66] A. Anand, E. Racah, S. Ozair, Y. Bengio, M.-A. Côté, and R. D. Hjelm, "Unsupervised state representation learning in atari," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 8769–8782.

[67] A. Stooke, K. Lee, P. Abbeel, and M. Laskin, "Decoupling representation learning from reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9870–9879.

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

[69] Z. D. Guo, M. G. Azar, A. Saade, S. Thakoor, B. Piot, B. A. Pires, M. Valko, T. Mesnard, T. Lattimore, and R. Munos, "Geometric entropic exploration," *arXiv preprint arXiv:2101.02055*, 2021.

[70] S. Nasiriany, V. Pong, S. Lin, and S. Levine, "Planning with goal-conditioned policies," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 14 814–14 825.

[71] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.