LIPSCHITZ-CONSTRAINED UNSUPERVISED SKILL DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

We study the problem of unsupervised skill discovery, whose goal is to learn a set of diverse and useful skills with no external reward. There have been a number of skill discovery methods based on maximizing the mutual information between skills and states. However, we point out that their objectives do not necessarily prefer dynamic skills that can be more useful for downstream tasks. To address this issue, we propose *Lipschitz-constrained Skill Discovery* (*LSD*), which encourages the agent to discover more diverse, dynamic, and far-reaching skills. Another benefit of LSD is that its learned representation function can be utilized for solving goal-reaching downstream tasks even in a zero-shot manner — *i.e.*, without further training or complex planning. Through experiments on various MuJoCo robotic locomotion and manipulation environments, we demonstrate that LSD outperforms previous approaches in terms of skill diversity, state space coverage, and performance on seven downstream tasks including the challenging task of following multiple goals on Humanoid. Videos of learned skills are available at https://sites.google.com/view/lipschitz-skill.

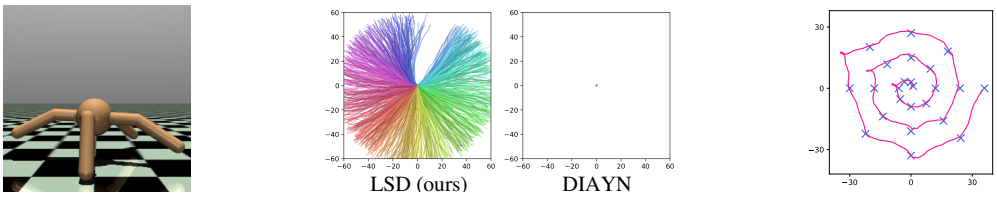
1 INTRODUCTION

Reinforcement learning (RL) aims at learning optimal actions that maximize accumulated reward signals (Sutton & Barto, 2018). Recently, RL with deep neural networks has demonstrated remarkable achievements in a variety of tasks, such as complex robotics control (Gu et al., 2017; Andrychowicz et al., 2020) and games (Schrittwieser et al., 2020; Badia et al., 2020). However, one limitation of the RL framework is that a practitioner has to manually define and tune a reward function for desired behaviors, which is often time-consuming and hardly scalable especially when there are multiple tasks to learn (Hadfield-Menell et al., 2017; Dulac-Arnold et al., 2019).

Therefore, several methods have been proposed to discover *skills* without external task rewards (Gregor et al., 2016; Eysenbach et al., 2019; Sharma et al., 2020), which is often referred to as the *unsupervised skill discovery* problem. Unsupervised discovery of skills helps not only relieve the burden of manually specifying a reward for each behavior, but also provide useful primitives to initialize with or combine hierarchically for solving downstream tasks (Eysenbach et al., 2019; Lee et al., 2020). Moreover, learned skills can effectively demonstrate the agent's capability in the environment, allowing a better understanding of both the agent and the environment.

One of the most common approaches to the unsupervised skill discovery problem is to maximize the mutual information (MI) between skill latent variables and states (Gregor et al., 2016; Achiam et al., 2018; Eysenbach et al., 2019; Hansen et al., 2020; Sharma et al., 2020; Choi et al., 2021; Zhang et al., 2021). Intuitively, these methods encourage a skill latent z to be maximally informative of states or trajectories obtained from a skill policy $\pi(a|s, z)$. As a result, optimizing the MI objective leads to the discovery of diverse and distinguishable behaviors.

However, existing MI-based skill discovery methods share a limitation that they do not necessarily prefer learning dynamic or task-relevant behaviors such as diverse locomotion skills. Since MI is invariant to scaling or any invertible transformation of the input variables, there exist infinitely many optima for the MI objective. As such, they will converge to the maximum that is most easily optimizable, which would usually be just learning simple and static skills. For instance, Figure 1b and Figure 12 demonstrate that DIAYN (Eysenbach et al., 2019) simply learns to take various postures in place rather than learning locomotion skills in the Ant environment (Schulman et al., 2015). While



(a) Ant environment. (b) Visualization of discovered 2-D skills on the x-y plane. (c) Zero-shot goal following.

Figure 1: Skills discovered by LSD can be used to follow goals with no further training or planning.

these works often employ some feature engineering or prior domain knowledge to discover more dynamic skills (*e.g.*, discriminating skills based on x-y coordinates only (Eysenbach et al., 2019; Sharma et al., 2020)), it brings about other drawbacks: (i) practitioners need to manually specify the dimensions of interest and (ii) the diversity of skills may be limited to a specific type (*e.g.*, the x-y prior results in neglecting non-locomotion behaviors).

In order to address this limitation of MI-based approaches, we propose an unsupervised skill discovery method based on a Lipschitz continuity constraint, named *Lipschitz-constrained Skill Discovery* (*LSD*). Specifically, we argue one reason behind the aforementioned problem is that the MI-based skill discovery methods can easily maximize the MI objective with only slight differences in the state space. To resolve this, we propose to formulate the objective as a constrained optimization problem with a Lipschitz constraint so that the objective maximization in the skill space entails an increase in state variations (Equation (6)).

LSD has several advantages. First, unlike previous MI-based skill discovery objectives, LSD encourages learned skills to have more variations in the state space, which leads to more diverse, dynamic, and far-reaching skills. Second, LSD produces a useful state representation function that can be used to solve goal-reaching downstream tasks in a fully *zero-shot* manner (Figure 1c); *i.e.*, with no further training or complex planning (such as simulation on learned models). Finally, LSD is easy to implement in contrast to many existing approaches and introduces no additional hyperparameters.

Our contributions can be summarized as follows:

- We propose LSD based on a Lipschitz constrained objective, which encourages the agent to discover skills with large state variations unlike existing MI-based approaches, and learns more diverse and dynamic behaviors with no need for feature engineering.
- The state representation function learned by LSD enables solving goal-reaching downstream tasks in a zero-shot fashion, with no burden of additional training or complex planning (Sharma et al., 2020). Table 1 highlights other distinctive properties of LSD in comparison to existing approaches.
- LSD exhibits the best performance in terms of the state space coverage on five MuJoCo environments and final rewards on seven downstream tasks, including AntMultiGoals (Kim et al., 2021), HumanoidMultiGoals and FetchPushGoal, compared to previous skill discovery methods such as DIAYN (Eysenbach et al., 2019), DADS (Sharma et al., 2020) and IBOL (Kim et al., 2021).

2 Preliminaries and Related Work

2.1 Problem Setting

We consider a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p)$ without external rewards, where $\mathcal{S}$ is a (continuous) state space, $\mathcal{A}$ is an action space, and $p(s_{t+1}|s_t, a_t)$ is a stochastic transition dynamics function. We represent a *skill* with a *latent variable* $z \in \mathcal{Z}$ and a latent-conditioned policy $\pi(a|s, z)$. The skill latent space $\mathcal{Z}$ can be either discrete or continuous; we use N to denote the number of skills in the discrete case, and d to denote the dimensionality of the skill latent space in the continuous case. Given a skill z and a skill policy $\pi(a|s, z)$, a trajectory $\tau = (s_0, a_0, \ldots, s_T)$ is sampled with the following generative process: $p^\pi(\tau|z) = p(s_0) \prod_{t=0}^{T-1} \pi(a_t|s_t, z) p(s_{t+1}|s_t, a_t)$. We use uppercase letters to denote random variables, and $h(\cdot)$ and $I(\cdot;\cdot)$ to represent the differential entropy and the mutual information, respectively.

Table 1: Comparison of unsupervised skill discovery methods. Refer to Section 2.2 for citations.

Property	VIC	DIAYN	DADS	VISR	EDL	APS	IBOL	LSD (ours)
Prefer 'dynamic' skills	✗	✗	✗	✗	✓	✓	✓	✓
Provide dense reward	✗	✓	✓	✓	✓	✓	✗	✓
Discover continuous skills	✓	✓	✓	✓	✓	✓	✓	✓
Discover discrete skills	✓	✓	✓	✗	✓	✗	✗	✓
Enable zero-shot goal-reaching	✗	✗	✓ †	✗	✗	✗	✗	✓

Properties — *"Prefer 'dynamic' skills"*: whether the algorithm prefers skills beyond simple and static ones. *"Enable zero-shot goal-reaching"*: whether learned skills can be used for goal-reaching downstream tasks without additional training, † denotes that it still needs planning.

2.2 Prior Work on Unsupervised Skill Discovery

A number of previous methods maximize $I(Z;S)$ to learn diverse skills. One line of research employs the identity $I(Z;S) = h(Z) - h(Z|S) \geq \mathbb{E}_{z\sim p(z), s\sim p^\pi(s|z)}[\log q(z|s)] - \mathbb{E}_{z\sim p(z)}[\log p(z)]$ or its variants, where the skill discriminator $q(z|s)$ is a variational approximation of the posterior $p(z|s)$ (Barber & Agakov, 2003). VIC (Gregor et al., 2016) maximizes the MI between the last states and skills given the initial state. DIAYN (Eysenbach et al., 2019) optimizes the MI between individual states and skills. VALOR (Achiam et al., 2018) also takes a similar approach, but considers the whole trajectories instead of states. VISR (Hansen et al., 2020) models the variational posterior as the von-Mises Fisher distribution, which results in an inner-product reward form and hence enables combining with successor features (Barreto et al., 2017). HIDIO (Zhang et al., 2021) examines multiple variants of the MI identity, and jointly learns skills with a hierarchical controller that maximizes the task reward. Choi et al. (2021) point out the equivalency between the MI-based objective and goal-conditioned RL, and show that Spectral Normalization (Miyato et al., 2018) improves the quality of learned skills. DADS (Sharma et al., 2020) maximizes the opposite direction of the mutual information identity $I(Z;S) = h(S) - h(S|Z)$ with the skill dynamics model $q(s_{t+1}|s_t, z)$, which allows zero-shot planning on downstream tasks. However, these methods share a limitation: they do not always prefer to reach distant states or to learn dynamic skills, as we can maximize $I(Z;S)$ even with the smallest state variations. One possible way to address this issue is to use heuristics such as feature engineering; for example, the x-y prior (Sharma et al., 2020) enforces skills to be discriminated only by their x-y coordinates so that the agent can discover locomotion skills.

On the other hand, a couple of methods overcome this limitation by integrating with exploration techniques. EDL (Campos Camúñez et al., 2020) first maximizes the state entropy $h(S)$ with SMM exploration (Lee et al., 2019), and then encodes the discovered states into skills via VAE (Kingma & Welling, 2014). APS (Liu & Abbeel, 2021a) combines VISR (Hansen et al., 2020) with APT (Liu & Abbeel, 2021b), an exploration method based on k-nearest neighbors. Yet, we empirically confirm that such a pure exploration signal is insufficient to make large and consistent transitions in states. IBOL (Kim et al., 2021) takes a hierarchical approach where it first pre-trains a low-level policy to make reaching remote states easier, and subsequently learns a high-level skill policy based on the information bottleneck framework (Tishby et al., 2000). While IBOL can discover skills reaching distant states in continuous control environments without locomotion priors (*e.g.*, x-y prior), it still has some limitations in that (1) IBOL cannot discover discrete skills, (2) it still capitalizes on input feature engineering in that they exclude the locomotion coordinates from the low-level policy, and (3) it consists of a two-level hierarchy with several additional hyperparameters, which make the implementation difficult. On the contrary, our proposed LSD can discover diverse skills in both discrete and continuous settings without using any feature engineering or extra prior knowledge, and is easy to implement as it requires no additional hyperparameters or hierarchy. Table 1 overviews the comparison of properties between different skill discovery methods.

3 Lipschitz-constrained Skill Discovery (LSD)

We first analyze limitations of existing MI-based methods for unsupervised skill discovery (Section 3.1), and then derive our approach for learning continuous skills, *Lipschitz-constrained Skill Discovery* (*LSD*), which encourages the agent to have large state variations (Section 3.2). We also show how learned skills can be used to solve goal-reaching tasks in a zero-shot fashion (Section 3.3), and extend LSD to discovery of discrete skills (Section 3.4).

3.1 LIMITATION OF MI-BASED METHODS

Before deriving our objective for discovery of continuous skills, we review variational MI-based skill discovery algorithms and discuss why such methods might end up with learning only simple and static skills. The MI objective $I(Z;S)$ with continuous skills can be written with the variational lower bound as follows (Eysenbach et al., 2019; Choi et al., 2021):

$$I(Z;S) = -h(Z|S) + h(Z) = \mathbb{E}_{z\sim p(z), s\sim p^{\pi}(s|z)}[\log p(z|s) - \log p(z)] \tag{1}$$

$$\geq \mathbb{E}_{z,s}[\log q(z|s)] + (\text{const}) = -\frac{1}{2}\mathbb{E}_{z,s}\left[\|z-\mu(s)\|^2\right] + (\text{const}), \tag{2}$$

where we assume that a skill $z \in \mathbb{R}^d$ is sampled from a fixed prior distribution $p(z)$, and $q(z|s)$ is a variational approximation of $p(z|s)$ (Barber & Agakov, 2003; Mohamed & Rezende, 2015), parameterized as a normal distribution with unit variance, $\mathcal{N}(\mu(s), \mathbf{I})$ (Choi et al., 2021). Some other methods are based on a conditional form of mutual information (Gregor et al., 2016; Sharma et al., 2020); for instance, the objective of VIC (Gregor et al., 2016) can be written as

$$I(Z;S_T|S_0) \geq \mathbb{E}_{z,\tau}[\log q(z|s_0,s_T)] + (\text{const}) = -\frac{1}{2}\mathbb{E}_{z,\tau}\left[\|z-\mu(s_0,s_T)\|^2\right] + (\text{const}), \tag{3}$$

where we assume that $p(z|s_0) = p(z)$ is a fixed prior distribution, and the posterior is chosen as $q(z|s_0,s_T) = \mathcal{N}(\mu(s_0,s_T),\mathbf{I})$ in a continuous skill setting.

One issue with Equations (2) and (3) is that these objectives can be maximized even with minuscule differences in states, since μ can be trained to map similar (yet different) states to different z's. This is especially problematic because discovering skills with such slight or less dynamic state variations is usually a 'lower-hanging fruit' than making dynamic and large differences in the state space (*e.g.*, μ can simply map the angles of Ant's legs or joints to z). As such, continuous DIAYN and DADS discover only posing skills on Ant (Figures 2a and 12) in the absence of any feature engineering or tricks to elicit more diverse behaviors.

3.2 CONTINUOUS LSD

In order to resolve the aforementioned issue, we will reformulate the VIC objective (Equation (3)) so that the maximization of our objective always entails an increase in state differences. In doing so, we will exploit the property of a Lipschitz constraint that it can force differences in the state space to be always larger than the corresponding differences in the latent space.

Further decomposition of the MI objective. Before introducing the constraint, we first model $\mu(s_0,s_T)$ in the VIC objective with $\phi(s_T)-\phi(s_0)$ to focus on the relative differences between the initial and final states, where $\phi : \mathcal{S} \to \mathcal{Z}$ is a learnable *state representation function* that maps a state observation into a latent space. This choice makes the latent skill z represent a direction or displacement in the latent space induced by ϕ. Then, we further decompose Equation (3) as follows:

$$\mathbb{E}_{z,\tau}\left[\log q(z|s_0,s_T)\right] + (\text{const}) = -\frac{1}{2}\mathbb{E}_{z,\tau}\left[\|z-(\phi(s_T)-\phi(s_0))\|^2\right] + (\text{const}) \tag{4}$$

$$= \underbrace{\mathbb{E}_{z,\tau}\left[(\phi(s_T)-\phi(s_0))^\top z\right]}_{\text{① alignment of the direction}} - \frac{1}{2}\underbrace{\mathbb{E}_{z,\tau}\left[\|\phi(s_T)-\phi(s_0)\|^2\right]}_{\text{② implicit regularizer}} + (\text{const}), \tag{5}$$

where we use the fact that $\mathbb{E}_z[z^\top z]$ is a constant as $p(z)$ is a fixed distribution. This decomposition of the MI lower-bound objective provides an intuitive interpretation: the first inner-product term ① in Equation (5) encourages the direction vector $\phi(s_T)-\phi(s_0)$ to be aligned with z, while the second term ② regularizes the norm of the vector $\phi(s_T)-\phi(s_0)$.

The LSD objective. Now, we employ a Lipschitz continuity constraint so that maximizing our objective in the latent space leads to an increase in state differences. Specifically, we suggest replacing the implicit regularizer of VIC (the second term in Equation (5)) with a 1-Lipschitz constraint on ϕ. This yields a constrained maximization objective only using the first term ① as follows:

$$J^{\text{LSD}} = \mathbb{E}_{z,\tau}\left[(\phi(s_T)-\phi(s_0))^\top z\right] \quad \text{s.t.} \quad \forall x,y \in \mathcal{S} \quad \|\phi(x)-\phi(y)\| \leq \|x-y\|, \tag{6}$$

where J^{LSD} is the objective of our proposed ***Lipschitz-constrained Skill Discovery*** (**LSD**). Note that LSD's objective differs from VIC's in an important way: Equation (4) tries to equate the value of z and $\phi(s_T)-\phi(s_0)$, while our objective Equation (6) only requires their directions to be aligned, which is especially advantageous as it enables zero-shot skill selection (Section 3.3).

The LSD objective encourages the agent to prefer skills with larger state variations, in contrast to VIC and other variants (*e.g.*, DIAYN, DADS), as follows. First, in order to maximize the inner product in Equation (6), the length of $\phi(s_T) - \phi(s_0)$ should be increased. It then makes its upper bound $\|s_T - s_0\|$ increase as well due to the 1-Lipschitz constraint (*i.e.*, $\|\phi(s_T) - \phi(s_0)\| \leq \|s_T - s_0\|$). As a result, it leads to learning more dynamic skills in terms of state differences.

Note that our purpose of enforcing the Lipschitz constraint is very different from its common usages in machine learning. Many works have adopted the Lipschitz continuity to regularize functions for better generalization (Neyshabur et al., 2018; Sokolic et al., 2017), interpretability (Tsipras et al., 2018) or stability (Choi et al., 2021). On the other hand, we employ it to ensure that maximization of the reward entails increased state variations. The Lipschitz constant 1 is chosen empirically as it can be easily implemented using Spectral Normalization (Miyato et al., 2018).

Per-step transition reward. By eliminating the second term in Equation (5), we can further decompose the objective using a telescoping sum as

$$J^{\text{LSD}} = \mathbb{E}_{z,\tau}\left[(\phi(s_T) - \phi(s_0))^\top z\right] = \mathbb{E}_{z,\tau}\left[\sum_{t=0}^{T-1}(\phi(s_{t+1}) - \phi(s_t))^\top z\right]. \tag{7}$$

This formulation enables the optimization of J^{LSD} with respect to the policy π (*i.e.*, reinforcement learning steps) with a per-step transition reward given as:

$$r_z^{\text{LSD}}(s_t, a_t, s_{t+1}) = (\phi(s_{t+1}) - \phi(s_t))^\top z. \tag{8}$$

Compared to the per-trajectory reward in Equation (3) (Gregor et al., 2016), it can be optimized more easily and stably with off-the-shelf RL algorithms such as SAC (Haarnoja et al., 2018a).

Connections to previous methods. The per-step reward function of Equation (8) is closely related to continuous DIAYN (Eysenbach et al., 2019; Choi et al., 2021) and VISR (Hansen et al., 2020)[1]:

$$r_z^{\text{DIAYN}}(s_t, a_t, s_{t+1}) = \log q^{\text{DIAYN}}(z|s_{t+1}) \propto -\|\phi(s_{t+1}) - z\|^2 + (\text{const}) \tag{9}$$

$$r_{\tilde{z}}^{\text{VISR}}(s_t, a_t, s_{t+1}) = \log q^{\text{VISR}}(\tilde{z}|s_t) \propto \tilde{\phi}(s_t)^\top \tilde{z} + (\text{const}) \tag{10}$$

$$r_z^{\text{LSD}}(s_t, a_t, s_{t+1}) = (\phi(s_{t+1}) - \phi(s_t))^\top z, \tag{11}$$

where $\tilde{z}$ and $\tilde{\phi}(s_t)$ in VISR are the normalized vectors of unit length (*i.e.*, $z/\|z\|$ and $\phi(s_t)/\|\phi(s_t)\|$), respectively. We assume that q^{DIAYN} is parameterized as a normal distribution with unit variance, and q^{VISR} as a von-Mises Fisher (vMF) distribution with a scale parameter of 1 (Hansen et al., 2020).

While it appears that there are some similarities among Equation (9)–(11), our LSD reward is fundamentally different from the others in that it optimizes neither a log-probability nor mutual information, and thus only LSD seeks for distant states. For instance, VISR (Equation (10)) has the most similar form as it also uses an inner product, but $\phi(s_{t+1}) - \phi(s_t)$ in the LSD reward does not need to be a unit vector unlike VISR optimizing the vMF distribution. Instead, LSD increases the difference of ϕ to encourage the agent to reach more distant states. We present an empirical comparison of the forms of the reward function in Appendix C.

Implementation. In order to maximize the LSD objective (Equation (8)), we alternately train π with SAC (Haarnoja et al., 2018a) and ϕ with stochastic gradient descent. We provide the full procedure for LSD in Appendix B. Note that LSD has the same components and hyperparameters as DIAYN, and is thus easy to implement, especially as opposed to IBOL (Kim et al., 2021), a state-of-the-art skill discovery method on MuJoCo environments that requires a two-level hierarchy with many moving components and hyperparameters.

3.3 Zero-shot Skill Selection

Another advantage of the LSD objective J^{LSD} (Equation (6)) is that the learned state representation $\phi(s)$ allows solving goal-reaching downstream tasks in a *zero-shot* manner (*i.e.*, without any further training or complex planning), as z is aligned with the direction in the representation space. Specifically, if we want to make a transition from the current state $s \in \mathcal{S}$ to a target state $g \in \mathcal{S}$, we can

[1]For DIAYN (Eysenbach et al., 2019), we omit the $-\log p(z)$ term from the original objective as its expectation can be treated as a constant assuming that $p(z)$ is a fixed prior distribution and that episodes have the same length. For VISR (Hansen et al., 2020), we only consider its unsupervised skill discovery objective.

simply repeat selecting the following z until reaching the goal:

$$z = \alpha(\phi(g) - \phi(s)) \,/\, \|\phi(g) - \phi(s)\|, \tag{12}$$

and executing the latent-conditioned policy $\pi(a|s, z)$ to choose an action. Here, α is a hyperparameter that controls the norm of z, and we find that $\alpha \approx \mathbb{E}_{z\sim p(z)}[\|z\|]$ empirically works the best (*e.g.*, $\alpha = 2^{-\frac{1}{2}}\Gamma(1/2) \approx 1.25$ for $d = 2$). As will be shown in Section 4.3, this zero-shot scheme provides a convenient way to immediately achieve strong performance on many goal-reaching downstream tasks. Note that, in such tasks, this scheme is more efficient than the zero-shot *planning* of DADS with its learned skill dynamics model (Sharma et al., 2020); LSD does not need to learn models or require any planning steps in the representation space.

3.4 Discrete LSD

Continuous LSD can be extended to discovery of *discrete* skills. One might be tempted to use the one-hot encoding for z with the same objective as continuous LSD (Equation (6)), as in prior methods (Gregor et al., 2016; Achiam et al., 2018; Eysenbach et al., 2019; Sharma et al., 2020). For discrete LSD, however, we cannot simply use the one-hot encoding, since it could make all the skills *collapse* into a single behavior. For example, a choice of $\phi(s_T) = [v, v, \dots, v]^\top$ and $\phi(s_0) = [0, 0, \dots, 0]^\top$ with some appropriate scalar v can trivially maximize J^{LSD} (Equation (6)). This failure mode stems from the fact that our objective does not pertain to a categorical distribution, unlike the previous works.

To resolve this issue, we propose to use the following *contrastive* one-hot encoding of z:

$$Z \sim \text{unif}\{z_1, z_2, \dots, z_N\} \quad \text{where } [z_i]_j = \begin{cases} 1 & \text{if } i = j \\ -\frac{1}{N-1} & \text{otherwise} \end{cases} \text{ for } i, j \in \{1, \dots, N\}, \tag{13}$$

where N is the number of skills, $[\cdot]_i$ denotes the i-th element of the vector, and $\text{unif}\{\dots\}$ denotes the uniform distribution over a set. Plugging into Equation (8), the reward for the k-th skill becomes

$$r_k^{\text{LSD}}(s_t, a_t, s_{t+1}) = [\phi(s_{t+1}) - \phi(s_t)]_k - \frac{1}{N-1} \sum_{i \in \{1,2,\dots,N\} \setminus \{k\}} [\phi(s_{t+1}) - \phi(s_t)]_i. \tag{14}$$

Intuitively, this objective enforces the k-th element of $\phi(s)$ to be the only indicator for the k-th skill, preventing the aforementioned collapse. Equation (14) consistently pushes the difference in ϕ to be as large as possible, unlike prior approaches using the softmax function (Eysenbach et al., 2019; Sharma et al., 2020). Thanks to the Lipschitz constraint, this makes the agent learn diverse and dynamic behaviors as in the continuous case.

4 Experiments

We compare LSD with multiple previous skill discovery methods on various MuJoCo robotic locomotion and manipulation environments (Todorov et al., 2012; Schulman et al., 2015; Plappert et al., 2018) from OpenAI Gym (Brockman et al., 2016). We aim to answer the following questions: (i) How well does LSD discover skills on high-dimensional continuous control problems, compared to previous approaches? (ii) Can the discovered skills be used for solving goal-reaching tasks in a zero-shot fashion? In Appendix, we present an ablation study for demonstrating the importance of LSD's components (Appendix C) and analyses of learned skills (Appendix D).

Experimental setup. We make evaluations on three MuJoCo robotic locomotion environments (Ant, Humanoid, HalfCheetah) and three robotic manipulation environments (FetchPush, FetchSlide and FetchPickAndPlace). Following the practice in previous works (Sharma et al., 2020; Kim et al., 2021), we mainly compare skill discovery methods on Ant, but we additionally adopt Humanoid for quantitative comparisons with competitive baselines, since it is often considered the most challenging environment in the MuJoCo benchmark. On the manipulation environments, we compare LSD to baselines combined with MUSIC-u (Zhao et al., 2021), an intrinsic reward that facilitates the agent to have control on target objects. For continuous skills, unless otherwise mentioned, we use two-dimensional skill latents ($d = 2$) sampled from the standard normal distribution, following Kim et al. (2021). On the locomotion environments, we normalize the state dimensions to ensure that the different scales of the dimensions have less effect on skill discovery. We repeat all the experiments eight times and denote their 95% confidence intervals with shaded areas or error bars. We refer to Appendix E for the full experimental details.

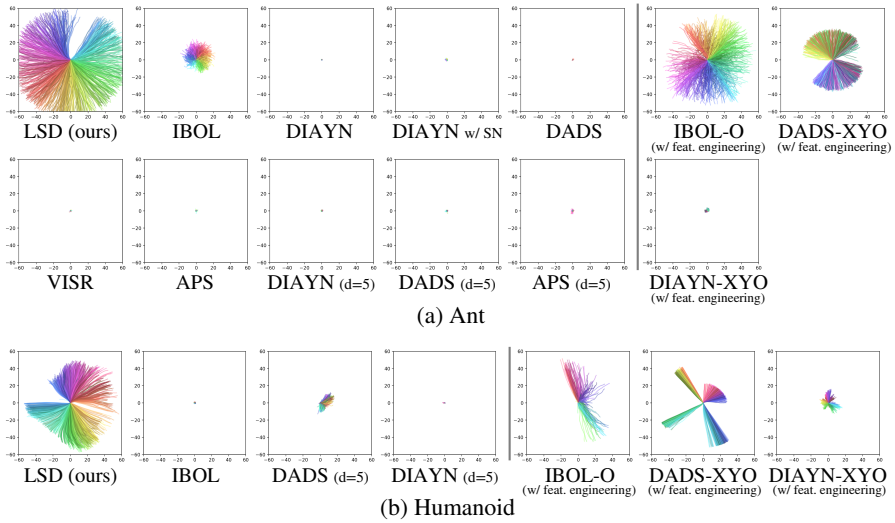


Figure 2: Visualization of 2-D (or 5-D) continuous skills discovered by various methods. We plot the x-y trajectories of the agent. Each color represents the direction of the skill latent variable z.

Baseline methods. We make comparisons with six skill discovery algorithms: DIAYN (Eysenbach et al., 2019), DADS (Sharma et al., 2020), VISR (Hansen et al., 2020), EDL (Campos Camúñez et al., 2020), APS (Liu & Abbeel, 2021a) and IBOL (Kim et al., 2021).

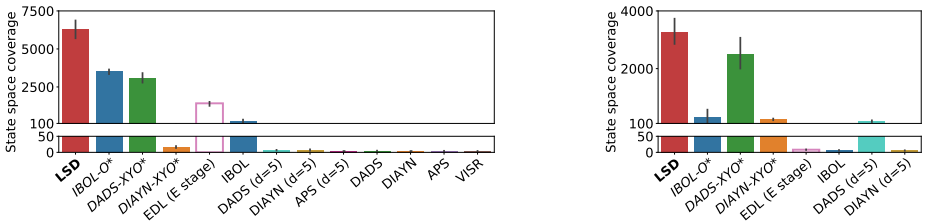
Additionally, we consider two variants of skill discovery methods: the x-y prior (denoted with the suffix '-XY') and x-y omission (with '-O') (Sharma et al., 2020; Kim et al., 2021). The x-y prior variant restricts the input to the skill discriminator or the dynamics model only to the positional information, enforcing the agent to discover locomotion skills. The x-y omission variant excludes the locomotion coordinates from the input to policies or dynamics models (but not to discriminators) to impose an inductive bias that the agent can choose actions regardless of its location. We denote the variants that have both modifications with the suffix '-XYO'. While previous methods mostly require such feature engineering or tricks to discover skills that move consistently or have large variations in the state space, we will demonstrate that LSD can discover diverse skills on MuJoCo locomotion environments with *no* such tricks.

4.1 Skills Learned with Continuous LSD

Visualization of skills. We train LSD and baselines on the Ant environment to learn two-dimensional continuous skills ($d = 2$). Figure 2a visualizes the learned skills as trajectories of the Ant agent on the x-y plane. LSD discovers skills that move far from the initial location in almost all possible directions, while the other methods except IBOL fail to discover such locomotion primitives without feature engineering (*i.e.*, x-y prior) even with an increased skill dimensionality ($d = 5$). Instead, they simply learn to take static postures rather than to move; such a phenomenon is also reported in Gu et al. (2021). This is because their MI objectives do not particularly induce the agent to increase state variations. On the other hand, LSD discovers skills reaching even farther than those of the baselines using feature engineering (IBOL-O, DADS-XYO and DIAYN-XYO).

Figure 2b demonstrates the results on the Humanoid environment. As in Ant, LSD learns diverse skills walking or running consistently in various directions, while skills discovered by other methods are limited in terms of the state space coverage or the variety of directions. Note that LSD, again, makes use of no feature engineering. We provide the videos of skills discovered by LSD at `https://sites.google.com/view/lipschitz-skill`.

Quantitative evaluation. For numerical comparison, we measure the *state space coverage* (Kim et al., 2021) of each skill discovery method on Ant and Humanoid. The state space coverage is measured by the number of occupied 1×1 bins on the x-y plane from 200 randomly sampled trajectories, averaged over eight runs. For EDL, we report the state space coverage of its SMM exploration phase (Lee et al., 2019). Figure 3 shows that on both environments, LSD outperforms all the baselines even including those with feature engineering.



(a) Ant (b) Humanoid

Figure 3: Plots of state space coverage. Asterisks (*) denote the methods with feature engineering.

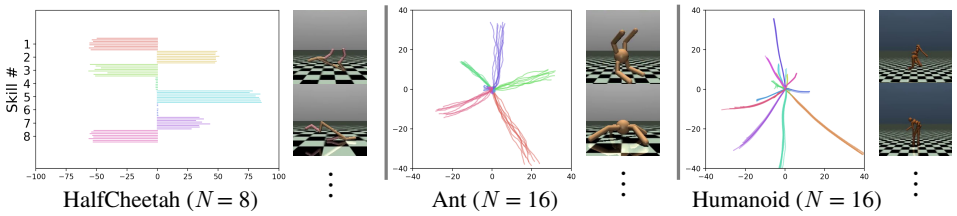
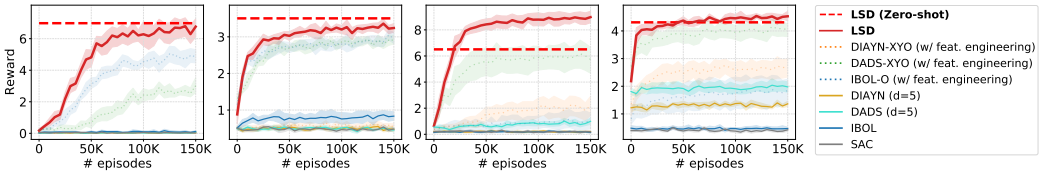


Figure 4: Qualitative results of discrete LSD (Section 4.2). We visualize each skill's trajectories on the x axis (HalfCheetah) or the x-y plane (Ant and Humanoid). See Appendix F for more results.



(a) AntGoal (b) AntMultiGoals (c) HumanoidGoal (d) HumanoidMultiGoals

Figure 5: Performance on downstream tasks after skill discovery (Section 4.3).

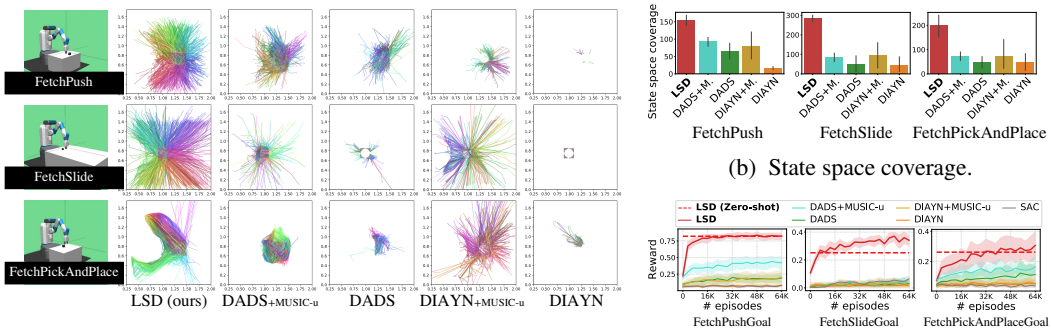
4.2 Skills Learned with Discrete LSD

We train discrete LSD on Ant, HalfCheetah and Humanoid with $N = 6, 8, 16$, where N is the number of skills. While continuous LSD mainly discovers locomotion skills, we observe that discrete LSD learns more diverse skills thanks to its contrastive learning scheme (Figure 4, Appendix F). On Ant, discrete LSD discovers a skill set consisting of five locomotion skills, six rotation skills, three posing skills and two flipping skills. On HalfCheetah, the agent learns to run forward and backward in multiple postures, to roll forward and backward, and to take different poses. Finally, Humanoid learns to run or move in multiple directions and speeds with unique gaits. We highly recommend the reader to check the videos available on our project page. To the best of our knowledge, LSD is the only method that can discover such diverse and dynamic behaviors (*i.e.*, having large variations in many of the state dimensions) within a single set of skills in the absence of feature engineering.

4.3 Comparison on Downstream Tasks

As done in Eysenbach et al. (2019); Sharma et al. (2020); Kim et al. (2021), we make comparisons on downstream goal-reaching tasks to assess how well skills learned by LSD can be employed for solving tasks in a hierarchical manner, where we evaluate our approach not only on AntGoal and AntMultiGoals (Eysenbach et al., 2019; Sharma et al., 2020; Kim et al., 2021) but also on the more challenging tasks: HumanoidGoal and HumanoidMultiGoals. In the '-Goal' tasks, the agent should reach a uniformly sampled random goal on the x-y plane, while in the '-MultiGoals' tasks, the agent should navigate through multiple randomly sampled goals in order. The agent is rewarded only when it reaches a goal. We refer to Appendix E for the full details of the tasks.

We first train each skill discovery method with continuous skills (without rewards), and then train a hierarchical meta-controller on top of the learned skill policy (kept frozen) with the task reward. The meta-controller observes the target goal concatenated to the state observation. At every K-th step, the controller selects a skill $z \in [-2, 2]^d$ to be performed for the next K steps, and the chosen skill is executed by the skill policy for the K steps. We also examine zero-shot skill selection of LSD, denoted 'LSD (Zero-shot)', where the agent chooses z at every step according to Equation (12).



(a) Visualization of trajectories of the target object on the x-y plane (*i.e.*, the table) in three Fetch manipulation environments. Each color represents the direction of the skill latent variable z.

(b) State space coverage.

(c) Average reward on downstream tasks.

Figure 6: Comparisons on the Fetch robotic manipulation environments (Section 4.4).

Results. Figure 5 shows the performance of each algorithm evaluated on the four downstream tasks. LSD demonstrates the highest reward in all of the environments, outperforming even the baselines with feature engineering. On top of that, in some environments such as AntMultiGoals, LSD's zero-shot skill selection performs the best, while still exhibiting strong performance on the other tasks. From these results, we show that LSD is capable of solving downstream goal-reaching tasks very efficiently with no further training or complex planning procedures.

4.4 Experiments on Robotic Manipulation Environments

In order to demonstrate that LSD can also discover useful skills in environments other than locomotion tasks, we make another evaluation on three Fetch robotic manipulation environments (Plappert et al., 2018). We compare LSD with other skill discovery methods combined with MUSIC-u (Zhao et al., 2021), an intrinsic reward that maximizes the mutual information $I(S^a; S^s)$ between the agent state S^a and the surrounding state S^s. For a fair comparison with MUSIC, we make use of the same prior knowledge for skill discovery methods including LSD to make them focus only on the surrounding states (Zhao et al., 2021), which correspond to the target object in our experiments.

Results. Figure 6a visualizes the target object's trajectories of 2-D continuous skills learned by each algorithm. They suggest that LSD can control the target object in the most diverse directions. Notably, in FetchPickAndPlace, LSD learns to pick the target object in multiple directions without any task reward or intrinsic motivation like MUSIC-u. Figures 6b and 6c present quantitative evaluations on the Fetch environments. As in Ant and Humanoid, LSD exhibits the best state space coverage (measured with 0.1×0.1-sized bins) in all the environments. Also, LSD and LSD (Zero-shot) outperform the baselines by large margins on the three downstream tasks.

5 Conclusion

We presented LSD as an unsupervised skill discovery method based on a Lipschitz constraint. We first pointed out the limitation of previous MI-based skill discovery objectives that they are likely to prefer static skills with limited state variations, and proposed to solve the issue by applying a Lipschitz constraint to our reformulated objective. It resulted in learning more dynamic skills and a state representation function that enables zero-shot skill selection, which is even more efficient than the zero-shot planning (Sharma et al., 2020) in goal-reaching tasks. Through multiple quantitative and qualitative experiments on robotic manipulation and complex locomotion environments, we showed that LSD outperforms previous skill discovery methods in terms of the diversity of skills, state space coverage and performance on downstream tasks.

Limitations and future directions. LSD may not encourage dynamic behaviors in some environments where Lipschitz constraints are not semantically meaningful, such as control from pixel observations. This issue is addressable by incorporating representation learning, which we leave for future work. Also, in contrast to discrete LSD that learns more diverse skills (thanks to its contrastive scheme), continuous LSD mainly discovers locomotion skills that move as far as possible, since its objective is independent of the magnitude of z. While this has an advantage in that we can later choose skills just by their directions (enabling zero-shot skill selection), making LSD respect the magnitude as well can be another interesting research direction.

REPRODUCIBILITY STATEMENT

We make our implementation publicly available in the anonymous repository at https://github.com/iclr2022lsd/LSD and provide the full implementation details in Appendix E.

REFERENCES

Joshua Achiam, Harrison Edwards, Dario Amodei, and Pieter Abbeel. Variational option discovery algorithms. *arXiv preprint arXiv:1807.10299*, 2018.

Marcin Andrychowicz, Bowen Baker, Maciek Chociej, R. Józefowicz, Bob McGrew, Jakub W. Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, S. Sidor, Joshua Tobin, P. Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39:20 – 3, 2020.

Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, P. Sprechmann, Alex Vitvitskyi, Daniel Guo, and C. Blundell. Agent57: Outperforming the atari human benchmark. *ArXiv*, abs/2003.13350, 2020.

D. Barber and F. Agakov. The IM algorithm: a variational approach to information maximization. In *Advances in Neural Information Processing Systems*, 2003.

André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado Van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

G. Brockman, Vicki Cheung, Ludwig Pettersson, J. Schneider, John Schulman, Jie Tang, and W. Zaremba. OpenAI Gym. *ArXiv*, abs/1606.01540, 2016.

Víctor Campos Camúñez, Alex Trott, Caiming Xiong, Richard Socher, Xavier Giró Nieto, and Jordi Torres Viñals. Explore, discover and learn: unsupervised discovery of state-covering skills. In *ICML*, 2020.

Jongwook Choi, Archit Sharma, Honglak Lee, Sergey Levine, and Shixiang Gu. Variational empowerment as representation learning for goal-conditioned reinforcement learning. In *ICML*, 2021.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5:613–624, 1993.

Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.

The garage contributors. Garage: A toolkit for reproducible reinforcement learning research. https://github.com/rlworkgroup/garage, 2019.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

Shixiang Gu, Ethan Holly, T. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3389–3396, 2017.

Shixiang Shane Gu, Manfred Diaz, C Daniel Freeman, Hiroki Furuta, Seyed Kamyar Seyed Ghasemipour, Anton Raichuk, Byron David, Erik Frey, Erwin Coumans, and Olivier Bachem. Braxlines: Fast and interactive toolkit for RL-driven behavior generation beyond reward maximization. 2021.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1861–1870. PMLR, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, G. Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *ArXiv*, abs/1812.05905, 2018b.

Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart Russell, and Anca Dragan. Inverse reward design. In *Advances in Neural Information Processing Systems*, 2017.

S. Hansen, Will Dabney, André Barreto, T. Wiele, David Warde-Farley, and V. Mnih. Fast task inference with variational intrinsic successor features. *ArXiv*, abs/1906.05030, 2020.

Jaekyeom Kim, Seohong Park, and Gunhee Kim. Unsupervised skill discovery with bottleneck option learning. In *ICML*, 2021.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014.

Lisa Lee, Benjamin Eysenbach, Emilio Parisotto, Eric Xing, Sergey Levine, and Ruslan Salakhutdinov. Efficient exploration via state marginal matching. *ArXiv*, abs/1906.05274, 2019.

Youngwoon Lee, Jingyun Yang, and Joseph J. Lim. Learning to coordinate manipulation skills via skill behavior diversification. In *International Conference on Learning Representations (ICLR)*, 2020.

Hao Liu and Pieter Abbeel. APS: Active pretraining with successor features. In *ICML*, 2021a.

Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. *ArXiv*, abs/2103.04551, 2021b.

Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. A laplacian framework for option discovery in reinforcement learning. *ArXiv*, abs/1703.00956, 2017.

Marlos C. Machado, C. Rosenbaum, Xiaoxiao Guo, Miao Liu, G. Tesauro, and Murray Campbell. Eigenoption discovery through the deep successor representation. *ArXiv*, abs/1710.11089, 2018.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *ArXiv*, abs/1802.05957, 2018.

Shakir Mohamed and Danilo J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. *arXiv preprint arXiv:1509.08173*, 2015.

Behnam Neyshabur, Srinadh Bhojanapalli, David A. McAllester, and Nathan Srebro. A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks. *ArXiv*, abs/1707.09564, 2018.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, Vikash Kumar, and Wojciech Zaremba. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *ArXiv*, abs/1802.09464, 2018.

Julian Schrittwieser, Ioannis Antonoglou, T. Hubert, K. Simonyan, L. Sifre, Simon Schmitt, A. Guez, Edward Lockhart, Demis Hassabis, T. Graepel, T. Lillicrap, and D. Silver. Mastering Atari, Go, Chess and Shogi by planning with a learned model. *Nature*, 588 7839:604–609, 2020.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *ArXiv*, abs/1506.02438, 2015.

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020.

Jure Sokolic, Raja Giryes, Guillermo Sapiro, and Miguel R. D. Rodrigues. Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, 65:4265–4280, 2017.

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

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. *ArXiv*, physics/0004057, 2000.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

D. Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and A. Madry. There is no free lunch in adversarial robustness (but there are unexpected benefits). *ArXiv*, abs/1805.12152, 2018.

Jesse Zhang, Haonan Yu, and Wei Xu. Hierarchical reinforcement learning by discovering intrinsic options. *ArXiv*, abs/2101.06521, 2021.

Rui Zhao, Yang Gao, Pieter Abbeel, Volker Tresp, and Wei Xu. Mutual information state intrinsic control. *ArXiv*, abs/2103.08107, 2021.

APPENDIX: LIPSCHITZ-CONSTRAINED UNSUPERVISED SKILL DISCOVERY

A EXTENDED RELATED WORK

The LSD reward (Equation (8)) might look very similar to the objective of *eigenoptions* (Machado et al., 2017; 2018): $(\phi(s_{t+1}) - \phi(s_t))^\top e$, where ϕ is a *fixed* (or pre-trained) representation function of states and e is an eigenvector of the successor representation matrix (Dayan, 1993) computed from a *fixed random* policy. They define eigenoptions as the options (or skills) that maximize this reward for each of the eigenvectors e of the N largest eigenvalues. However, our interest differs from their setting, since we *learn* both the policy and the representation function in order to seek diverse and dynamic skills, and our approach is applicable to continuous skill settings as well as discrete skill learning.

B TRAINING PROCEDURE FOR LSD

Algorithm 1: Lipschitz-constrained Skill Discovery (LSD)

Initialize skill policy π and representation function ϕ;
while *not converged* **do**
  for $i = 1, \dots,$ *(# episodes per epoch)* **do**
    Sample skill z from $p(z)$;
    Sample trajectory (episode) τ with $\pi(\cdot|\cdot, z)$ and z;
  Compute reward $r^{\text{LSD}}(s_t, a_t, s_{t+1}) = (\phi(s_{t+1}) - \phi(s_t))^\top z$ (Equation (8));
  Update ϕ using SGD to maximize Equation (8) under Spectral Normalization;
  Update π using SAC;

Algorithm 1 overviews the training procedure for LSD. There are two learnable components in LSD: the skill policy $\pi(a|s, z)$ and the representation function $\phi(s)$. In order to impose the 1-Lipschitz constraint on ϕ, we employ Spectral Normalization (SN) (Miyato et al., 2018). At every epoch, we alternately train π with SAC (Haarnoja et al., 2018a) and ϕ with stochastic gradient descent (SGD) to maximize Equation (8). When collecting trajectories τ, we fix z within a single episode as in previous works (Eysenbach et al., 2019; Sharma et al., 2020; Kim et al., 2021).

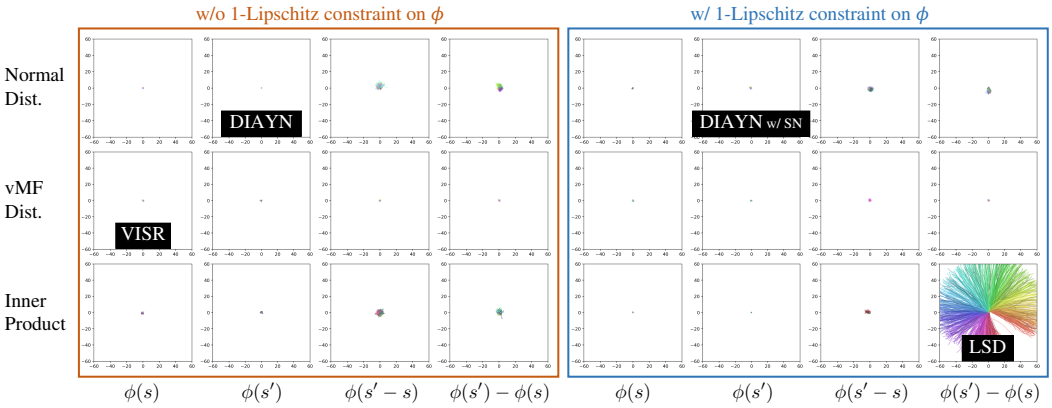


Figure 7: Ablation study on Ant. See Appendix C.

C Ablation Study

The LSD reward (Equations (6) and (8)) consists of three components: the inner-product form, the use of ϕ's difference and the 1-Lipschitz constraint on ϕ. In order to examine the role and importance of each component, we ablate them with several variants. We consider all possible combinations of the following variations:

(i) **The form of the reward function** specifying how z and $\phi(\cdot)$ are compared:
(1) the Normal distribution form (as in DIAYN, Equation (9)), *i.e.*, the squared distance,
(2) the von-Mises Fisher (vMF) distribution form (as in VISR, Equation (10)), *i.e.*, the inner-product form with normalizing the norms of $\phi(\cdot)$ and z, or
(3) the inner-product form (as in LSD, Equations (8) and (11)) without normalizing the norms.

(ii) **The use of the current and/or next states**: $\phi(s)$, $\phi(s')$ or $\phi(s'-s)$ in place of $\phi(s')-\phi(s)$ (where s' denotes the next state).

(iii) **The use of Spectral Normalization**: with or without the 1-Lipschitz constraint on $\phi(\cdot)$.

Figure 7 shows the result of the ablation study on the Ant environment. We observe the state space coverage drastically decreases if any of these components constituting the LSD reward (Equation (8)) is missing; *i.e.*, all of the three components are crucial for LSD. We especially note that just adding Spectral Normalization on DIAYN ('DIAYN w/ SN', Choi et al. (2021)) does not sufficiently induce state variations, since its objective does not necessarily encourage the scale of $\phi(s)$ to increase.

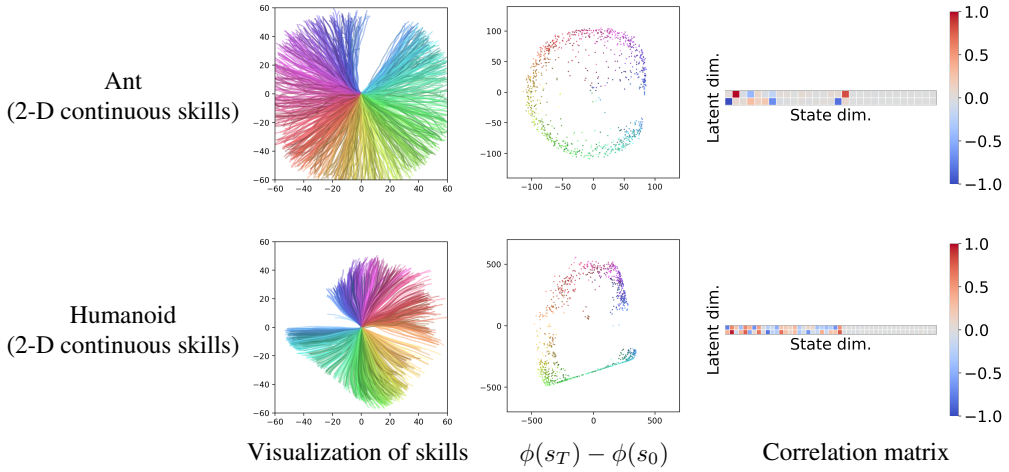


Figure 8: Analysis of 2-D continuous skills for Ant and Humanoid.

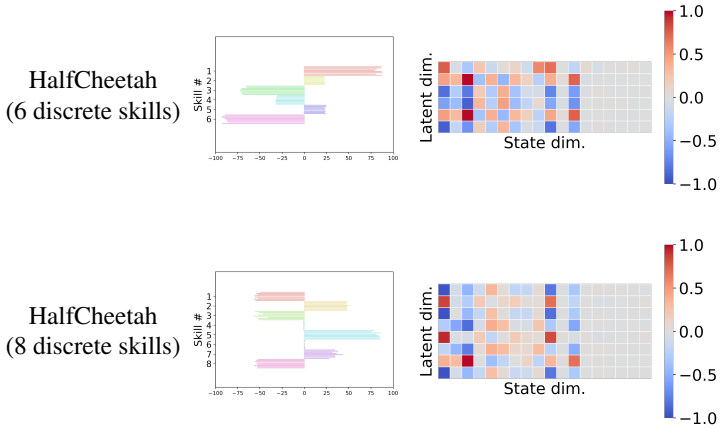


Figure 9: Analysis of discrete skills for HalfCheetah.

D Visualization and Analyses of Skills Learned

We provide more visual examples and analyses of skills discovered by LSD (Figures 8 to 11, also videos available at https://sites.google.com/view/lipschitz-skill).

Figure 8 visualizes 2-D continuous skills for Ant and Humanoid, and the learned state representation function ϕ, and demonstrates the correlation coefficient matrices between the state dimensions and skill latent dimensions. We observe that continuous LSD focuses on the x-y coordinates (the first and second state dimensions) on both the environments, which is attributable to that those two dimensions are the most suitable ones for increasing $\phi(s_T) - \phi(s_0)$ under the Lipschitz constraint. Figure 8 also shows that ϕ on Ant has almost no correlation with the 8th to 15th state dimensions, which correspond to the angles of the four leg joints. This is because Ant should repeatedly move its legs back and forth to move consistently in a direction.

Figures 9 to 11 illustrate discrete skills learned for HalfCheetah, Ant and Humanoid. We confirm that discrete LSD focuses not only on the x-y coordinates but on a set of more diverse dimensions. For instance, the example of 'Ant (16 discrete skills)' in Figure 10 shows that some skills such as the second and third ones have large correlations with the orientation dimensions (the 4th to 7th state dimensions) of Ant. These skills correspond to rotation skills, as shown in the video on our project page.

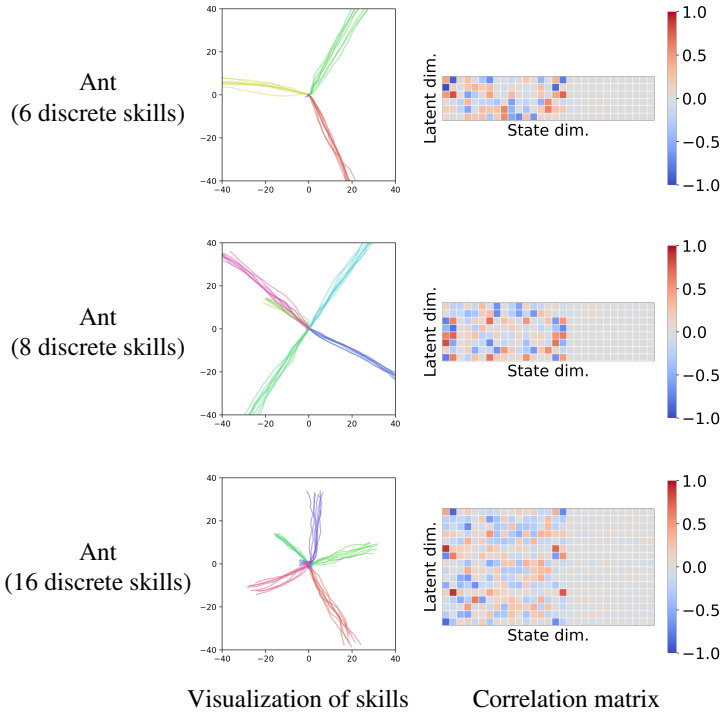


Figure 10: Analysis of discrete skills for Ant.

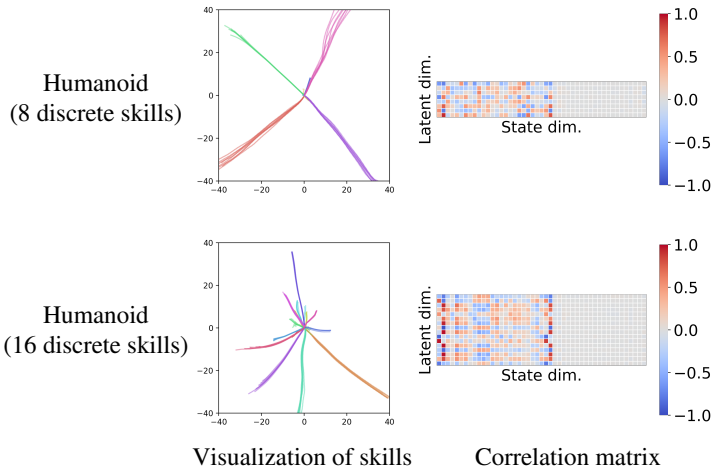


Figure 11: Analysis of discrete skills for Humanoid.

E IMPLEMENTATION DETAILS

E.1 MUJOCO LOCOMOTION ENVIRONMENTS

E.1.1 SETTINGS

We implement LSD and most of the baselines on top of the garage framework (garage contributors, 2019), while we train the E stage (Lee et al., 2019) of EDL (Campos Camúñez et al., 2020) using their released codebase[2]. We provide the implementation for LSD in the anonymous repository at https://github.com/iclr2022lsd/LSD.

For the environments used in our experiments, we use the same configurations adopted in Sharma et al. (2020); Kim et al. (2021), with a maximum episode length of 200 for Ant and HalfCheetah, and 1000 for Humanoid. We normalize each state dimension of the environments ahead of skill discovery with the statistics used in Kim et al. (2021), where they compute the mean and standard deviation from 10000 roll-outs of a random policy.

E.1.2 VARIANTS OF SKILL DISCOVERY METHODS

DIAYN-XYO (Eysenbach et al., 2019) limits the input to the skill discriminator $q(z|s)$ to the x-y dimensions and omits them from the input to its skill policy. For DADS-XYO (Sharma et al., 2020), we make its skill dynamics model $q(s'|s, z)$ only consider the x-y coordinates and exclude them from the inputs to both the skill policy and the skill dynamics model, as done in Sharma et al. (2020). IBOL-O (Kim et al., 2021) denotes the exact setting used in their work, where they omit the x-y coordinates from the input to its low-level policy.

E.1.3 TRAINING OF SKILL DISCOVERY METHODS

We model each trainable component as a two-layered MLP with 1024 units, and train them with SAC (Haarnoja et al., 2018a). At every epoch, we sample 20 (Ant, HalfCheetah) or 5 (Humanoid) roll-outs and train the networks with 4 gradient steps computed from 2048-sized mini-batches. For quantitative evaluation of learned skills, we train the models for 2M episodes (= 100K epochs, Ant) or 1M episodes (= 200K epochs, Humanoid).

For each method, we search the discount factor γ from $\{0.99, 0.995\}$ and the SAC entropy coefficient α from $\{0.003, 0.01, 0.03, 0.1, 0.3, 1.0, \text{auto-adjust}\}$ (Haarnoja et al., 2018b). For continuous skills on Ant, we use $\gamma = 0.995$ for the low-level policy of IBOL-O and 0.99 for the others, and use auto-adjusted α for DADS, DIAYN, VISR and APS, $\alpha = 0.01$ for LSD, $\alpha = 0.03$ for DADS ($d = 5$), and $\alpha = 0.3$ for DIAYN ($d = 5$). We set the default learning rate to $1e-4$, but $3e-5$ for DADS's q, DIAYN's q and LSD's ϕ, and $3e-4$ for IBOL's low-level policy. On Ant and HalfCheetah, we train the models with on-policy samples without using the replay buffer (following Sharma et al. (2020); Kim et al. (2021)), while we use the replay buffer for sampling the $k(= 5)$-nearest neighbors in APS. For the low-level policy of IBOL-O, we additionally normalize rewards and use full-sized batches, following Kim et al. (2021). In the case of IBOL, we used mini-batches of size 1024 and do not normalize rewards, as we find this setting performs better. We use the original hyperparameter choices for their high-level policies.

On Humanoid, we set the discount factor to $\gamma = 0.99$ and the learning rate to $3e-4$, but $1e-4$ for DADS's q, DIAYN's q and LSD's ϕ. Also, we use the replay buffer and additionally give an alive bonus b at each step (following Kim et al. (2021)) searched from $\{0, 0.03, 0.3\}$, while we find that b does not significantly affect the performance. We use $\alpha = 0.03$ for LSD and auto-adjusted α for the others, and $b = 0$ for DADS and $b = 0.03$ for the others.

E.1.4 TRAINING OF SMM

We train the SMM exploration stage (Lee et al., 2019) of EDL (Campos Camúñez et al., 2020) using the official implementation with hyperparameters tuned. Specifically, we set the discount factor γ to 0.99 (Ant, Humanoid), the SAC entropy coefficient α to 1 (Ant) or 3 (Humanoid), the β-VAE coefficient to 1, the alive bonus b to 30 (only for Humanoid), and the density coefficient ν to 0.5 (Ant) or 0.05 (Humanoid).

[2] https://github.com/victorcampos7/edl

E.1.5 DOWNSTREAM TASKS

In AntGoal and HumanoidGoal, a random goal $g \in [-g_s, g_s]^2$ is given at the beginning of each episode, and the episode ends if the agent reaches the goal (*i.e.*, the distance between the agent and the goal becomes less than or equal to ϵ). In AntMultiGoal and HumanoidMultiGoals, (up to X) new goals are sampled within the relative range of $[-g_m, g_m]^2$ from the current 2-D coordinates when the episode begins, the agent reaches the current goal, or the agent does not reach the goal within Y steps. In all of the environments, the agent receives a reward of r when it reaches the goal (no reward otherwise). We use $g_s = 50$, $g_m = 15$, $\epsilon = 3$, $X = 4$ and $Y = 50$ for AntGoal and AntMultiGoals, and $g_s = 20$, $g_m = 7.5$, $\epsilon = 3$, $X = 4$ and $Y = 250$ for HumanoidGoal and HumanoidMultiGoals. Also, we set r to 10 ('-Goals' environments) or 2.5 ('-MultiGoals' environments).

E.1.6 TRAINING OF HIERARCHICAL POLICIES FOR DOWNSTREAM TASKS

For downstream tasks, we train a high-level meta-controller on top of a pre-trained skill policy, where the meta-controller is modeled as an MLP with two hidden layers of 512 dimensions. We set K to 25 (Ant) or 125 (Humanoid), the learning rate to $3e-4$, the discount factor to 0.995, and use the auto-adjusted entropy coefficient. At each epoch, we sample ten trajectories and train the networks with four SAC gradients computed from full-sized batches. We normalize the state dimensions with an exponential moving average. For zero-shot learning, we set g in Equation (12) to the current state with its locomotion dimensions replaced with the goal's coordinates.

E.2 MUJOCO MANIPULATION ENVIRONMENTS

E.2.1 TRAINING OF SKILL DISCOVERY METHODS

For MuJoCo manipulation environments (FetchPush, FetchSlide, FetchPickAndPlace), we implement skill discovery methods based on the official implementation[3] of MUSIC (Zhao et al., 2021). We train each method for 8K episodes (= 4K epochs) with SAC and set the model dimensionality to $(1024, 1024)$, the entropy coefficient to 0.02, the discount factor to 0.95 and the learning rate to 0.001. At every epoch, we sample two trajectories and train the models with 40 gradient steps computed from 256-sized mini-batches. For methods equipped with the MUSIC intrinsic reward, we set the MUSIC reward coefficient to 5000 with reward clipping, following Zhao et al. (2021). For the skill reward coefficient, we perform hyperparameter search over $\{5, 15, 50, 150, 500, 1500, 5000\}$, where we choose 500 (LSD), 150 (DADS), or 1500 (DIAYN), not clipping the skill reward.

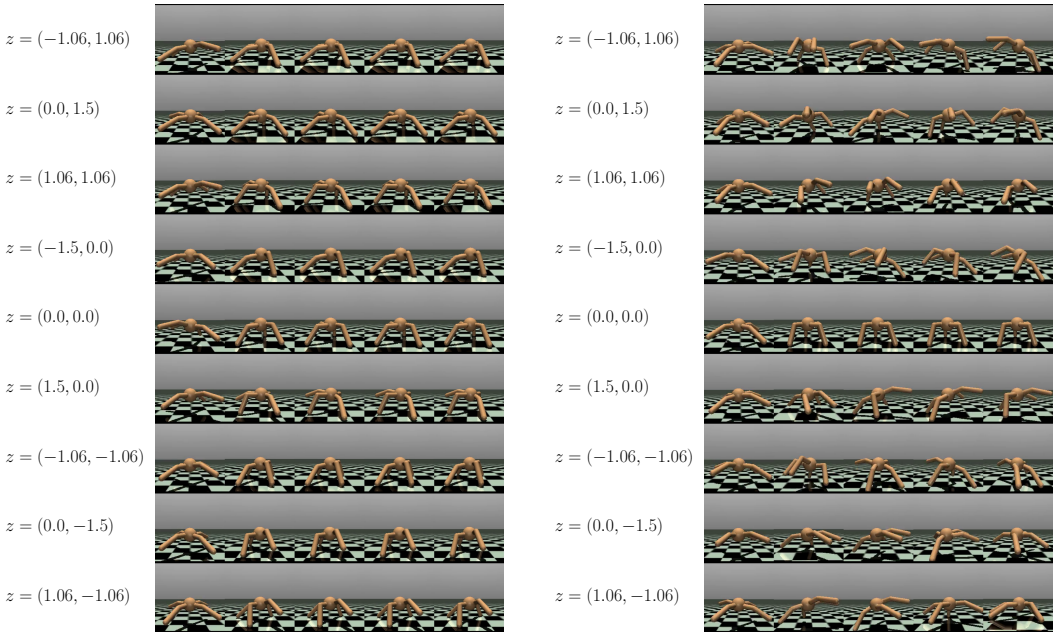
E.2.2 TRAINING OF DOWNSTREAM POLICIES

In FetchPushGoal, FetchSlideGoal and FetchPickAndPlaceGoal, a random goal is sampled at the beginning of each episode and the episode ends with a reward of 1 if the agent reaches the goal, where we use the same sampling range and reach radius as the original Fetch environments. For training of meta-controllers, we use the same hyperparameters as in the skill discovery phase, except that we sample 16 trajectories for each epoch.

F RENDERED SCENES OF LEARNED SKILLS

Figures 12 to 16 visualize the rendered scenes of skills discovered on the MuJoCo locomotion environments. Each figure demonstrates a *single* set of skills. We refer the reader to our project page for the videos.

[3] https://github.com/ruizhaogit/music



(a) DIAYN (b) LSD

Figure 12: 2-D continuous skills for Ant. DIAYN discovers posing skills, as its mutual information objective does not necessarily prefer large state variations. On the other hand, LSD encourages the agent to have more variations in the state space, resulting in learning more dynamic behaviors such as locomotion skills. Videos are available on our project page.

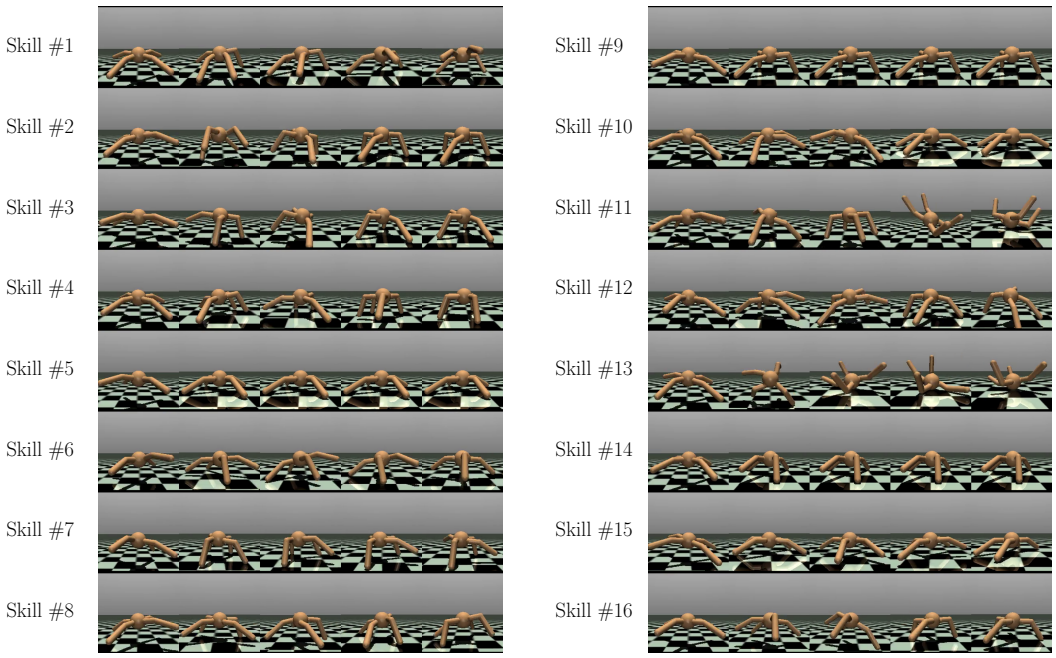


Figure 13: 16 discrete skills discovered by LSD for Ant. Discrete LSD learns a skill set consisting of locomotion skills (#1, #6, #7, #12, #16), rotation skills (#2, #3, #4, #8, #10, #15), posing skills (#5, #9, #14) and flipping skills (#11, #13). Videos are available on our project page.

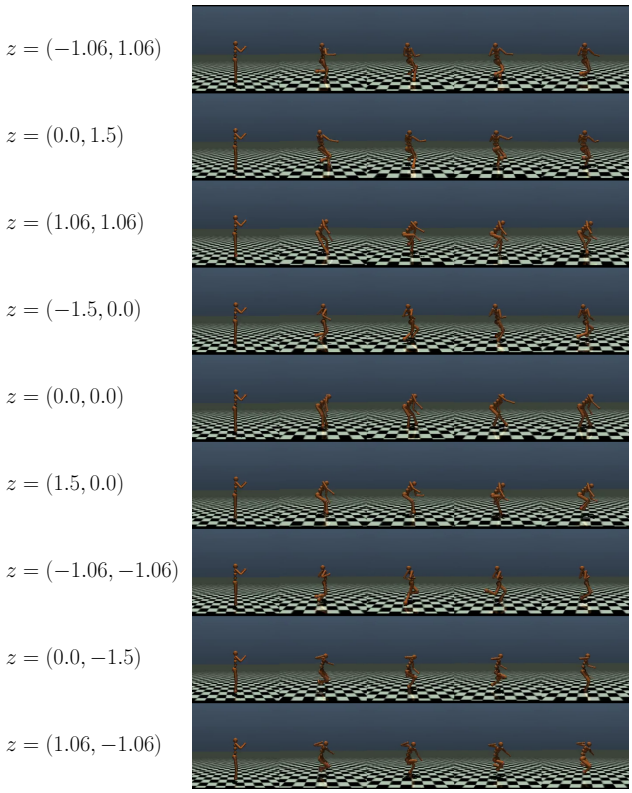


Figure 14: 2-D continuous skills discovered by LSD for Humanoid. The Humanoid robot can walk in various directions specified by z. Videos are available on our project page.

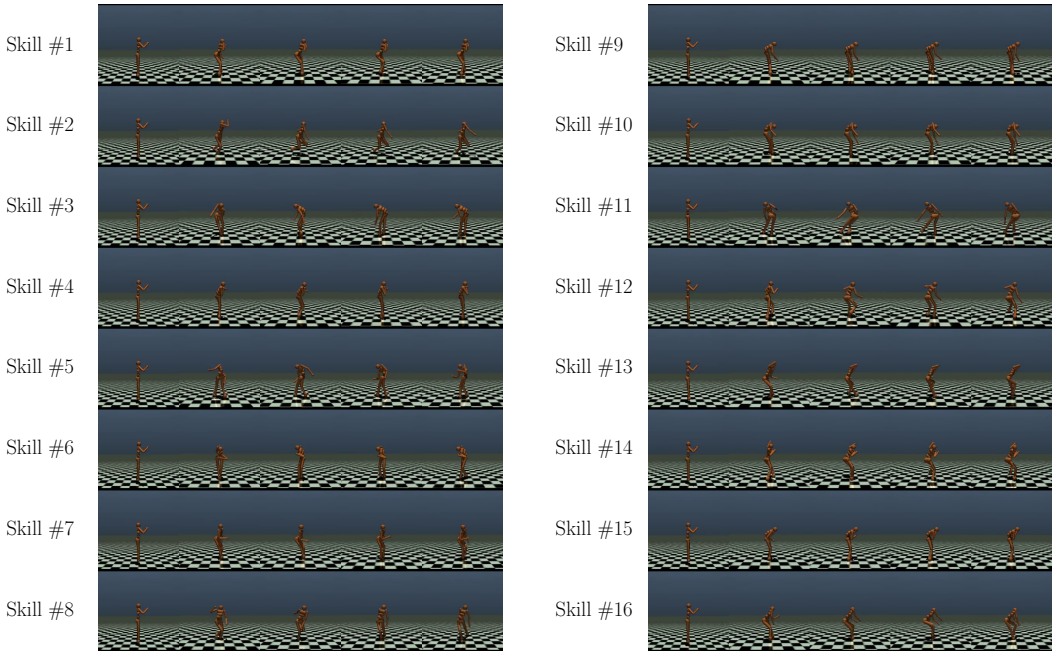


Figure 15: 16 discrete skills discovered by LSD for Humanoid. Videos are available on our project page.

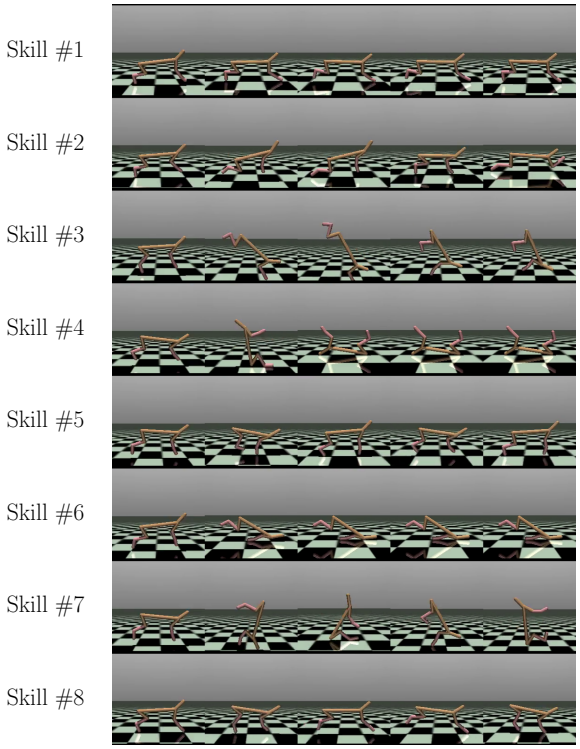


Figure 16: 8 discrete skills discovered by LSD for HalfCheetah. Videos are available on our project page.