GUIDING SKILL DISCOVERY WITH FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Unsupervised skill discovery removes the need for hand-designed rewards, but maximizing behavioral diversity alone readily produces undesirable or dangerous skills: a cheetah robot learns to roll in every direction when we would prefer it to run without flipping or entering hazardous areas. We propose **Fo**undation model **G**uided (FoG) skill discovery, which incorporates such preferences without reward functions or demonstrations. A preference is stated once, in language; a foundation model turns it into a binary label on states zero-shot, by writing the labeler as a short program from a description of the state vector or by scoring raw pixels against the sentence. Starting from a trajectory-level Wasserstein-dependency formulation, we show how this label enters the skill-discovery reward as a multiplicative gate through the critic of the Kantorovich-Rubinstein dual, and prove that the resulting objective lower-bounds the trajectory-level dependency. Across state- and pixel-based benchmarks, FoG eliminates flipping and rolling, avoids hazardous areas, discovers behaviors that are hard to specify explicitly, and yields skill libraries that adapt quickly and safely to downstream goals. Interactive visualisations are available from `https://sites.google.com/view/submission-fog`.

1 INTRODUCTION

Reinforcement Learning (RL) has achieved striking results in robotics and gaming (Tang et al., 2024; Vasco et al., 2024), but its deployment is bottlenecked by reward engineering (Sowerby et al., 2022). Unsupervised RL (Laskin et al., 2021) sidesteps this bottleneck by letting agents discover a diverse repertoire of skills without any task reward, to be adapted or composed downstream, making it an attractive route to general-purpose pretraining of decision-making agents (Park et al., 2023c).

Unrestricted exploration, however, is misaligned with practical needs. Robotics (Singh et al., 2022), agentic assistants (Bai et al., 2022), and household robots (Soni et al., 2024) require agents that respect human intentions and avoid behaviors that are unsafe, costly, or uninteresting; a cheetah that learns to flip and roll has diverse skills of little use to a user who cares about locomotion (Park et al., 2023c). Since skill discovery is pretraining whose output is a library of skills, unrestricted diversity spends much of that library on behaviors no downstream task wants, some of which damage hardware before any task is attempted. What is needed is not a task reward, which would specify one behavior and collapse the library to it, but a preference: a coarse statement of where in state space skills are worth discovering, applied once and shared by every downstream task. Preferences such as "do not flip" or "look like running" are cheap to state in language and expensive to convert into a reward or demonstrations (Kim et al., 2024).

Skill discovery has become increasingly metric-aware. Mutual-information objectives (Eysenbach et al., 2018; Sharma et al., 2019) reward statistically distinguishable skills but are blind to how far apart they move the agent, and often produce static behaviors (Park et al., 2022; 2023c). Replacing KL with the 1-Wasserstein distance yields the Wasserstein Dependency Measure (WDM) (Ozair et al., 2019), which rewards skills for being far apart under a chosen metric; METRA (Park et al., 2023c) instantiates it with a temporal distance

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

and achieves state-of-the-art coverage. None of these engage with user intent. Foundation models can score a state against a sentence zero-shot; what has been missing is a principled way to fold such a signal into the discovery objective so that diversity remains the driving force and the preference only decides which diversity counts. We use foundation models in the narrowest way that makes this work: as a zero-shot labeler mapping a state to a yes/no judgment about a sentence, with no fine-tuning, no per-step querying of a language model, and no environment-in-the-loop refinement. Our contributions are:

- **A formulation** of preference-aligned skill discovery in which the preference is a multiplicative gate on the diversity reward rather than an additive reward term, contrasted term by term with demonstration-based (Kim et al., 2024) and reward-based alternatives (Appendix C).
- **A derivation.** We lift the WDM from states to trajectories and show how the gate enters through the critic of its Kantorovich-Rubinstein dual; the per-step objective lower-bounds the trajectory-level dependency and reduces to METRA when the gate is constant.
- **FoG**, an instantiation with off-the-shelf foundation models that reaches language-specified behaviors ("twisted", "running") with no hand-coded definition, with an analysis of the design choices that make the zero-shot signal usable.
- **Validation** against eight baselines on state- and pixel-based benchmarks, including downstream goal-reaching with the discovered skills, with an open-source codebase.

2 Problem Setting and Preliminary

We work with a controlled Markov process, an MDP without a reward: $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mu, p)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, initial-state distribution $\mu \in \Delta(\mathcal{S})$, and transition kernel $p : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$. We introduce latent vectors $z \in \mathcal{Z}$ (discrete or continuous) and a latent-conditioned policy $\pi(a \mid s, z)$, and call each z with the policy it induces a skill. A trajectory is generated by drawing $z \sim p(z)$, holding it fixed, and rolling out $\pi(a \mid s, z)$ under p, so that

$$P_\pi(\tau, z) = p(z)\,\mu(s_0)\prod_{t=0}^{T-1}\pi(a_t \mid s_t, z)\,p(s_{t+1} \mid s_t, a_t), \tag{1}$$

where $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$. The goal is a family $\{\pi(\cdot \mid \cdot, z)\}_{z\in\mathcal{Z}}$ of diverse and useful behaviors learned without supervision, demonstrations, or hand-crafted priors.

The metric-aware alternative to mutual information is the Wasserstein Dependency Measure (Park et al., 2023c),

$$I_{\mathcal{W}}(S; Z) = \mathcal{W}\big(p(s, z),\, p(s)\,p(z)\big), \tag{2}$$

where $\mathcal{W}$ is the 1-Wasserstein distance on $(\mathcal{S} \times \mathcal{Z},\, d)$. WDM rewards skills for being far apart under d, which makes the metric central: Euclidean distance suffices for low-dimensional states but fails for images, so designing d is a core modeling choice (Park et al., 2025; 2022; 2023b;c; Rho et al., 2024).

3 Preference-Gated Skill Discovery via Trajectory Wasserstein Dependency

Metric-aware skill discovery (Park et al., 2023c) learns a state representation ϕ and rewards a skill z for moving through it in the direction z; how far a skill moves is the only source of diversity. FoG asks instead how far a skill moves through states the user would accept: the per-step diversity credit is multiplied by a gate $w(s_{t+1}) \in \{\alpha, 1\}$ that a foundation model sets zero-shot from a single sentence (Algorithm 1). This section shows that the gate is not a heuristic. It falls out of the Wasserstein-dependency view once the dependency is measured on trajectories rather than states (Sections 3.1 and 3.2), it comes with a preference-aware metric

Algorithm 1 FoG. The labeler c is obtained once; the training loop is METRA with a gated reward.

Require: intention sentence(s); foundation model; skill prior $p(z)$; METRA backbone (π, ϕ); α; stride N

1: **Once:** obtain $c : \mathcal{S} \to \{0, 1\}$ from the foundation model (Section 3.4)
2: Sample $z \sim p(z)$; roll out $\tau = (s_0, \ldots, s_T)$ with $\pi(\cdot \mid s, z)$
3: Label every Nth state with c (reusing it for the next $N - 1$); set $w(s_{t+1}) = \alpha + (1 - \alpha)\, c(s_{t+1})$
4: $r_t \leftarrow w(s_{t+1}) \big(\phi(s_{t+1}) - \phi(s_t)\big)^\top z$; update ϕ by equation 9 and π on r_t as in METRA

under which the gated objective provably lower-bounds the trajectory-level dependency (Section 3.3), and it admits a zero-shot instantiation (Section 3.4).

3.1 From states to trajectories

Preferences are properties of whole behaviors (staying upright throughout an episode, never entering a region), so we extend the Wasserstein dependency framework (Park et al., 2023c) from states to trajectories. With $P_\pi(\tau, z) = p(z)\, P_\pi(\tau \mid z)$, marginal $P_\pi(\tau) = \int P_\pi(\tau \mid z)\, p(z)\, dz$, and a metric $d_{(\tau,z)}$ on $\mathcal{T} \times \mathcal{Z}$, we measure how strongly behavior depends on the skill by

$$\mathrm{WDM}(\pi) \;:=\; W_1^{(d_{(\tau,z)})}\big(P_\pi(\tau, z),\; P_\pi(\tau)\, p(z)\big). \tag{3}$$

A large $\mathrm{WDM}(\pi)$ means different skills induce trajectory distributions far apart under $d_{(\tau,z)}$; the metric decides what counts as diverse. Given finite first moments, the Kantorovich-Rubinstein (KR) dual (Villani et al., 2009; Ozair et al., 2019) gives

$$\mathrm{WDM}(\pi) = \sup_{\|f\|_{\mathrm{Lip}} \le 1} \mathbb{E}_{P_\pi(\tau,z)}[f(\tau, z)] - \mathbb{E}_{P_\pi(\tau)\,p(z)}[f(\tau, z)], \tag{4}$$

over critics f that are 1-Lipschitz with respect to $d_{(\tau,z)}$. Any admissible critic lower-bounds $\mathrm{WDM}(\pi)$, and the critic family is where preferences enter.

3.2 A preference-gated critic

Following Park et al. (2023c), we restrict the critic to the bilinear form $f_\psi(\tau, z) = \langle \Phi_\psi(\tau), z \rangle, \Phi_\psi(\tau) \in \mathbb{R}^d, z \in \mathbb{R}^d$, and define the trajectory embedding additively over transitions,

$$\Phi_\psi(\tau) = \sum\nolimits_{t=0}^{T-1} w(s_{t+1}) \big(\phi(s_{t+1}) - \phi(s_t)\big). \tag{5}$$

where $\phi : \mathcal{S} \to \mathbb{R}^d$ is a learned state representation and $w(s_{t+1})$ gates each transition. Each transition contributes its displacement in representation space scaled by the desirability of the state it enters, so the preference acts step by step; w depends on s_{t+1} alone because preferences are expressed over outcomes, which keeps the specification to one label per state. Expanding,

$$f_\psi(\tau, z) = \sum\nolimits_{t=0}^{T-1} z^\top (\phi(s_{t+1}) - \phi(s_t))\, w(s_{t+1}). \tag{6}$$

so the critic measures the alignment between z and the preference-weighted motion of τ.

We take the gate to be binary, $w(s) \in \{\alpha, 1\}$ with $0 < \alpha \ll 1$, parameterized by a label $c : \mathcal{S} \to \{0, 1\}$:

$$w(s) = \alpha + (1 - \alpha)\, c(s)$$

so that transitions into states with $c(s) = 0$ are downweighted by α. The discrete form matches what foundation models produce natively, a yes/no judgment, whereas a calibrated real-valued score would import

prompt-phrasing and calibration noise into every reward; Section 4.4 confirms that the binary gate beats raw similarity scores and tolerates heavy corruption of c.

Since a fixed 1-Lipschitz critic never exceeds the supremum in equation 4,

$$\mathrm{WDM}(\pi) \;\geq\; \mathcal{L}(\pi,\phi) := \mathbb{E}_{P_\pi(\tau,z)}\big[\langle \Phi_\psi(\tau), z\rangle\big] - \mathbb{E}_{P_\pi(\tau)\,p(z)}\big[\langle \Phi_\psi(\tau), z\rangle\big], \tag{7}$$

provided f_ψ is 1-Lipschitz (enforced in Section 3.3). By linearity in z, the second term equals $\mathbb{E}_{P_\pi(\tau)}\big[\langle \Phi_\psi(\tau),\, \mathbb{E}_{p(z)}[z]\rangle\big]$ and vanishes for a centered prior, $\mathbb{E}_{p(z)}[z] = \bar{z} = 0$, which we assume without loss of generality. The bound collapses to

$$\mathcal{L}(\pi,\phi) = \mathbb{E}_{z\sim p(z),\,\tau\sim P_\pi(\cdot|z)}\big[\langle \Phi_\psi(\tau), z\rangle\big],$$

whose summands $w(s_{t+1})(\phi(s_{t+1}) - \phi(s_t))^\top z$ are the per-step intrinsic rewards. When $w(\cdot) \equiv 1$ the sum telescopes to $\phi(s_T) - \phi(s_0)$ and $\mathcal{L}$ is exactly METRA's objective, which METRA derives by specializing the state-level dependency to $I_\mathcal{W}(S_T; Z)$ and applying $\phi(s_T) - \phi(s_0) = \sum_t (\phi(s_{t+1}) - \phi(s_t))$ (Park et al., 2023c, Eq. (6)). A non-constant $w(s_{t+1})$ breaks that identity, so the state-level derivation cannot host a gate; the trajectory-level start is what makes gating well-defined.

3.3 Lipschitz constraint and lower-bound guarantee

The bound equation 7 needs f_ψ 1-Lipschitz on $\mathcal{T} \times \mathcal{Z}$. As in METRA (Park et al., 2023c), we split the constraint: $\|z\|_2 \leq 1$ makes $z \mapsto \langle \Phi_\psi(\tau), z\rangle$ Lipschitz whenever $\|\Phi_\psi(\tau)\|_2$ is bounded, and Φ_ψ is required to be Lipschitz under a trajectory metric, which is tight up to constants and easy to enforce. We lift the temporal shortest-path distance d_{temp} (Park et al., 2023c; Hartikainen et al., 2019; Durugkar et al., 2021), which is representation-agnostic and scales to pixels. Because w jumps by $1-\alpha$ across the preference boundary while adjacent states can straddle it, we inflate distances across the boundary by $\beta > 0$,

$$d_{\tilde{s}}(s,s') = d_{\mathrm{temp}}(s,s') + \beta\,\mathbf{1}\{c(s) \neq c(s')\}, \qquad d_{\tilde{\tau}}(\tau,\tau') = \sum\nolimits_{t=0}^{T} d_{\tilde{s}}(s_t, s'_t). \tag{8}$$

making w Lipschitz with constant $(1-\alpha)/\beta$ (Lemma B.1). Since $d_{\mathrm{temp}}(s,s') = 1$ on adjacent pairs $\mathcal{S}_{\mathrm{adj}}$ and shortest paths are chains of such steps, a local constraint on $\mathcal{S}_{\mathrm{adj}}$ is equivalent to the global one (Lemma B.2). Collecting the pieces,

$$\sup_{\pi,\phi}\; \mathbb{E}_{z\sim p(z),\,\tau\sim P_\pi(\cdot|z)}\left[\sum_{t=0}^{T-1} w(s_{t+1})\big(\phi(s_{t+1}) - \phi(s_t)\big)^\top z\right] \text{ s.t. } \|\phi(s) - \phi(s')\|_2 \leq 1 \;\; \forall (s,s') \in \mathcal{S}_{\mathrm{adj}}. \tag{9}$$

which is METRA's problem with the gate inserted into the reward. The following theorem licenses the insertion: every feasible (π,ϕ) certifies a constant-factor lower bound on the trajectory-level dependency, so maximizing equation 9 is a variational route to maximizing $\mathrm{WDM}(\pi)$.

Theorem 3.1 (Preference-gated WDM lower bound)**.** *Let π be a skill-conditioned policy with skill prior $p(z)$ supported on $\{z \in \mathbb{R}^d : \|z\|_2 \leq 1\}$ and centered ($\mathbb{E}_{p(z)}[z] = 0$). Let $w(s) \in \{\alpha, 1\}$ with $0 < \alpha \leq 1$ and label $c : \mathcal{S} \to \{0,1\}$ as in equation 5, and let $d_{\tilde{\tau}}$ be the hybrid trajectory metric in equation 8 with $\beta > 0$. Suppose $\phi : \mathcal{S} \to \mathbb{R}^d$ satisfies*

$$\|\phi(s) - \phi(s')\|_2 \leq 1 \;\; \forall (s,s') \in \mathcal{S}_{\mathrm{adj}}, \qquad \|\phi(s)\|_2 \leq B_\phi \;\; \forall s \in \mathcal{S},$$

and let $L_\Phi := 2 + \frac{2B_\phi(1-\alpha)}{\beta}$ and $\lambda := T/L_\Phi$. Define the joint metric $d_{(\tau,z)}\big((\tau,z),(\tau',z')\big) := d_{\tilde{\tau}}(\tau,\tau') + \lambda\,\|z - z'\|_2$. Then the inner expectation in equation 9 satisfies

$$W_1^{(d_{(\tau,z)})}\big(P_\pi(\tau,z),\; P_\pi(\tau)p(z)\big) \;\geq\; \frac{1}{L_\Phi}\,\mathbb{E}_{z\sim p(z),\,\tau\sim P_\pi(\cdot|z)}\left[\sum_{t=0}^{T-1} w(s_{t+1})\big(\phi(s_{t+1}) - \phi(s_t)\big)^\top z\right].$$

Proof sketch. The proof (Appendix B.2) chains KR duality equation 4, Lipschitzness of the rescaled critic $\tilde{f}_\psi = L_\Phi^{-1}\langle \Phi_\psi, z\rangle$ under the hybrid metric (Lemma B.1), the local-to-global equivalence (Lemma B.2), and the centered prior. The slack comes only from the bilinear critic family, componentwise Lipschitz control, and $\|\phi(s)\|_2 \le B_\phi$. □

3.4 Instantiating the gate with foundation models

The interface FoG asks of a foundation model is small: given a state and a sentence, return yes or no. The model is used off the shelf, queried once to build the labeler, and never called per step during training. Code is in the supplementary material.

For state vectors, a language model (ChatGPT or Claude; Appendix G.6) writes the labeler. We give it the environment's own description of the task and of the observation and action vectors (each dimension with its meaning, as in the Gym documentation; Appendix G.7.1) and one instruction, verbatim for HalfCheetah: `According to the given info, could you please write a python function to check if the cheetah is flipped over or not. If yes, output 1 otherwise 0.` The returned few-line function (Figure 2, top) is used unchanged as c, with no execution feedback and no iteration; unlike Eureka (Ma et al., 2023), which refines a complete reward function over rounds of environment feedback, FoG's model only has to recognize a condition.

For pixels, we use CLIP (Radford et al., 2021), a vision-language model with a shared image-text embedding space. The user supplies one sentence for the desirable condition and one for the undesirable one (for Cheetah flipping: `'The simulated two-leg robot stands normally'` and `'... flips over'`; all sentences are in Appendix G.7.2), and each frame is labeled by the sentence it is closer to,

$$w(s) = \alpha + (1-\alpha)\,\mathbf{1}[\cos(E_s, E_{t_1}) > \cos(E_s, E_{t_2})]\,, \tag{10}$$

where E_s, E_{t_1}, E_{t_2} embed the state, desirable description, and undesirable description, respectively. We embed every Nth frame and reuse its label for the next $N-1$; the cost is CLIP forward passes, about $1.5\times$ METRA's wall-clock at $N=2$ (Appendix G.4), with no extra environment interactions. $\alpha = 0$ attempts to not learn undesirable behaviors at all; $\alpha = 1$ recovers METRA. Appendix C contrasts the gated objective with demonstration- and reward-based alternatives term by term.

4 Experiments



Figure 1: Tasks used. HalfCheetah and Ant are state-based; the other three are pixel-based.

We ask: (1) Can a language model write a usable gate from a description of the state vector (Section 4.1)? (2) From pixels alone, can a vision-language model gate discovery well enough to suppress unsafe behaviors, avoid regions, and elicit behaviors with no hand-coded definition (Section 4.2)? (3) Are the resulting skills more useful, and safer to adapt, downstream (Section 4.3)? (4) Which design choices for the gate matter (Section 4.4)?

Baselines and protocol. We compare against eight baselines: **METRA** (Park et al., 2023c) and **LSD** (Park et al., 2022) (pure unsupervised discovery); **DoDont** (Kim et al., 2024), which gates METRA with a classifier trained on expert videos labeled with the ground-truth criterion, **DoDont+**, the same with videos annotated by the foundation model, and **LGSD** (Rho et al., 2024), which encodes intent as a language distance over privileged simulator state; and **FR-SAC** (foundation-model score as the sole SAC reward), **METRA+FR** (score added to the METRA reward), and **METRA+** (METRA gated by a hand-coded score). DoDont and METRA+ thus use privileged information that FoG does not. Appendix G.2 gives details and Appendix C writes each objective next to FoG's. Every method receives the same environment interactions per task (Table 3); FoG's foundation-model calls add compute (Appendix G.4) but no interactions. Results are means over three seeds with standard errors; $\alpha = 0$ and $N = 2$ unless stated. Videos are on the project website[1].

4.1 State-based Tasks

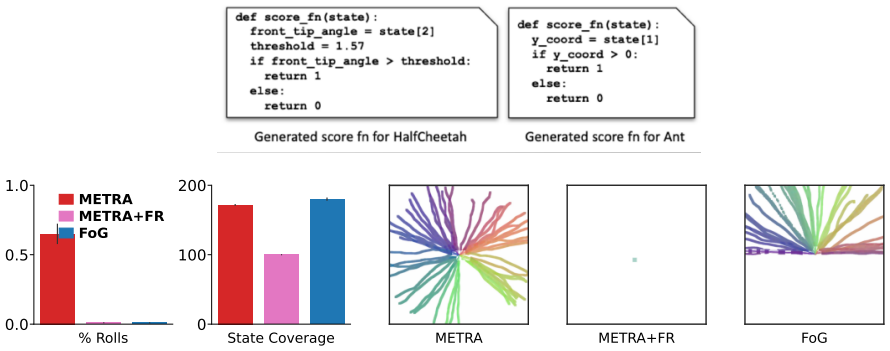

Figure 2: **Top:** the labeler written by the language model from one query; it picks the relevant state dimension (front-tip angle in HalfCheetah, y-coordinate in Ant) and a threshold. **Bottom left:** roll rate and state coverage in HalfCheetah. **Bottom right:** trajectories of learned skills in Ant. FoG never rolls and keeps METRA's coverage; METRA+FR neither rolls nor explores.

On HalfCheetah and Ant we ask the language model for a gate that avoids flipping and avoids moving south, respectively. It identifies the right state dimension and threshold in both cases (angle > 1.57 rad, $y > 0$; Figure 2, top). Gating METRA with these functions, FoG eliminates rolling in HalfCheetah (METRA rolls in about two thirds of episodes) at METRA's coverage, and moves north in Ant where METRA moves in all directions. Adding the same score to METRA's reward (METRA+FR) instead halves coverage in HalfCheetah and leaves the Ant agent static.

4.2 Pixel-based Tasks

In visual tasks where only images are available as observations, the gate comes from CLIP (Radford et al., 2021) via Equation (10).

Eliminating undesirable behaviors. In Cheetah we use `'agent flips over'` versus `'agent stands normally'`. Five methods retain METRA's full coverage of about 29 cells (METRA, DoDont, DoDont+, LGSD, FoG), and among them FoG has by far the fewest flips: about 12% of episodes, against 75% for METRA, 35% for DoDont, 52% for DoDont+, and 25% for LGSD (Figure 3). METRA+, LSD, FR-SAC,

[1] https://sites.google.com/view/submission-fog

and METRA+FR flip in under 2% of episodes only because they barely move (coverage ≤ 4). METRA+ suffers because a good hand-coded score ($r_{run} - r_{flip}$ per Kim et al. (2024)) is nontrivial; DoDont's classifier exploits ground color and scores unseen flipping states as "do" despite ground-truth labels (Appendix E); LGSD can only bring flipped states close together in skill space, not remove them; FR-SAC learns nothing useful, showing the score alone is not a trainable reward (Appendix F.5, Appendix C).

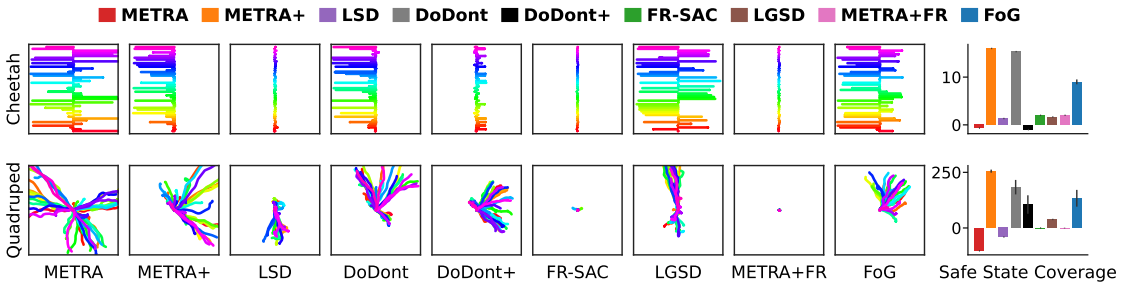

Figure 4: Avoiding hazardous areas. **Top**: Cheetah, skills by x-coordinate over time. **Bottom**: Quadruped, skills by xy-coordinates. Right: safe-state coverage (mean $\pm$ standard error). The privileged methods (METRA+, DoDont) cover the safe region best; FoG is next and best among methods without privileged information.

Avoiding hazardous areas. We designate the right area of Cheetah and the bottom-left of Quadruped as hazardous and, with no explicit directional cue, express the intent through the floor color (e.g., `'ground is blue'` vs. `'ground is orange'`). Figure 4 shows a three-way split. METRA+ and DoDont attain the highest safe-state coverage in both tasks, as they should: METRA+ is gated by the true region indicator and DoDont's demonstrations are labeled with it. FoG comes next in both tasks and is the best of the methods that work from the sentence alone. The rest fail: METRA explores indiscriminately, LSD, FR-SAC, and METRA+FR barely move, LGSD shrinks the hazardous region in skill space without keeping skills out of it, and DoDont+, whose demonstrations are annotated by the same CLIP model FoG uses ($\sim 70\%$ label accuracy), covers little of the safe region. That last comparison is the controlled one: given the same foundation model, gating discovery directly is far more robust to its errors than training a classifier on its labels.

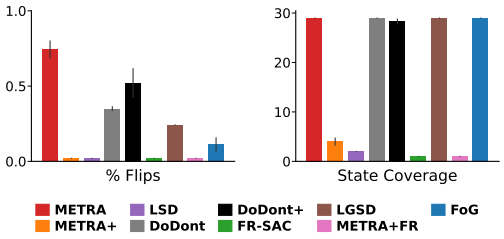

Figure 3: Avoiding flips in Cheetah: flip fraction (left) and state coverage (right) of the learned skills. FoG flips rarely while keeping METRA's coverage. Mean $\pm$ standard error.

Behaviors without a hand-coded definition. Humanoid (21D actions) is where a preference like "twisted" has no obvious reward. With `'agent is stretched'` vs. `'agent is twisted'`, FoG exhibits visibly more twisted postures while METRA is more stretched (Figure 5, left); ten participants shown paired videos chose FoG as more twisted 90% of the time (Appendix G.5). In a "Puppet" variant where the humanoid is held upright from above (Appendix G.1) and asked for running postures, FoG runs in all directions while METRA wriggles in squat postures, and participants preferred FoG 70% of the time.

4.3 Downstream Utility of the Discovered Skills

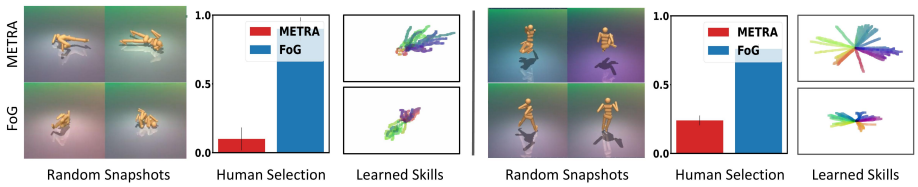

Figure 5: METRA vs. FoG on Humanoid (left) and Puppet (right): random snapshots, human selection rate, and skills by xy-coordinates. Participants judged FoG more "twisted" (90%) and more "running" (70%).

Following Park et al. (2023c), we freeze each method's skills and train a SAC (Haarnoja et al., 2018) controller that selects a skill every K low-level steps. The downstream problem is a distribution of goal-reaching tasks, one per skill library of Figures 3 and 4: each episode draws a goal and the controller is rewarded on reaching it, with goals restricted to the safe region in the two hazard tasks. Controllers are trained for equal interactions on top of libraries that consumed equal budgets, so Table 1 compares at matched total cost. We report the return over the adaptation window and at its end, and a safety score measuring how often the controller flips or enters the hazardous region while it learns.

Table 1: Downstream goal reaching with frozen skills over three tasks (Cheetah not-flip, Cheetah area, Quadruped area). Return AUC: mean normalised return over the first 5×10^4 controller steps; Final: normalised return at the end of that window; Safety: $1-$ flip fraction (not-flip) or hazard occupancy (area) while adapting. Best in bold; † privileged information.

Method	Return AUC ↑	Final return ↑	Safety ↑
FoG	**0.79** ±**0.03**	**0.88** ±**0.01**	0.94 ±0.02
METRA	0.74 ±0.02	0.85 ±0.01	0.81 ±0.03
LGSD	0.69 ±0.03	0.75 ±0.01	0.80 ±0.04
DoDont†	0.69 ±0.02	0.81 ±0.00	0.96 ±0.00
METRA+†	0.60 ±0.02	0.68 ±0.01	**0.97** ±**0.01**
DoDont+	0.57 ±0.01	0.65 ±0.02	0.75 ±0.01
LSD	0.17 ±0.01	0.19 ±0.01	0.76 ±0.02
METRA+FR	0.11 ±0.01	0.12 ±0.03	0.87 ±0.00

Three patterns emerge. First, FoG's skills give the best controller on return, during adaptation and at convergence, ahead of METRA and every other method without privileged information; on Quadruped, where adaptation speed discriminates most, FoG adapts fastest among them. Second, the libraries differ in what they contain, and this shows up as safety: FoG's has no flipping skill to select, so its controller flips an order of magnitude less often than METRA's and spends far less time in hazardous regions. Only the two privileged methods are safer, and both pay for it in adaptation because their libraries are more conservative. Third, gating is what makes the foundation model usable: DoDont+, which trains a classifier on the same CLIP labels, falls below METRA on every column, the static libraries (LSD, METRA+FR) give poor controllers, and a flat SAC agent without skills reaches only a small fraction of the return within the same budget. The setup assumes the discovery preference is shared by the downstream tasks, which is exactly the sense in which a preference differs from a task reward.

4.4 Design Choices for the Foundation-Model Gate

The gate is the only new component, so we examine the three choices that set its quality (Figure 6). Gate floor α: raising α from 0 to 0.5 and 0.8 steadily raises the flip rate, and replacing the hard gate by the raw similarity score (**sim**) brings it close to ungated METRA, so a binary gate is necessary, not merely convenient; $\alpha = 0$ can over-constrain exploration in some tasks (Appendix F.1). Label noise: flipping the label with probability b leaves the flip rate unchanged up to $b = 0.1$ and degrades it gracefully beyond, with coverage at METRA's level throughout, consistent with the gate scaling rather than replacing the diversity reward. Foundation model: SigLIP (Zhai et al., 2023) in place of CLIP with the same sentences more than doubles the flip rate, so the labeler matters more than the gate's hyperparameters. Appendix F.3 reports two further

choices, the labeling stride N and the phrasing of the intention; the latter is the main practical cost of the zero-shot interface, since some synonyms suppress flips only by suppressing motion, and we report every sentence used.

5 Related Work

Mutual-information skill discovery maximizes $I(S;Z) = H(S) - H(S|Z)$ (Eysenbach et al., 2018; Sharma et al., 2019; Laskin et al., 2022) and can be satisfied by static skills (Park et al., 2023c; 2022); SASD (Kim et al., 2023) and EDL (Hussonnois et al., 2023) add human preferences but are limited to state inputs and often need manual feedback. Distance-maximizing skill discovery (DSD) (Park et al., 2023b) maximizes traveled distance under a metric: LSD (Park et al., 2022) and CSD (Park et al., 2023b) use Euclidean distance and state density, METRA (Park et al., 2023c) a temporal distance that extends to pixels, PSD (Park et al., 2025) a periodic manifold, LGSD (Rho et al., 2024) text-embedding distances, and DoDont (Kim et al., 2024) a classifier learned from demonstrations (Table 2). FoG combines the temporal distance with a foundation-model gate in the hybrid metric equation 8, enabling zero-shot intent specification. It is closest to DoDont, which shares the multiplicative gate (Appendix C) but needs demonstrations that are costly (Fu et al., 2024) or impossible for "twisted" poses, and to LGSD, which is limited to low-dimensional tasks by rule-based state-to-text conversion and step-wise LLM queries. Foundation models as reward generators (Eureka (Ma et al., 2023), Motif (Klissarov et al., 2023), LAMP (Adeniji et al., 2023)) are discussed in Appendix D.

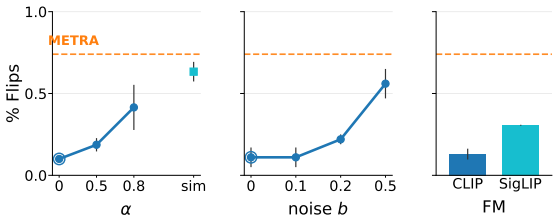

Figure 6: Gate design choices in Cheetah (flip fraction; dashed: METRA; hollow: default). Flips grow with the floor α and with label noise b; "sim" uses the raw similarity score; CLIP is a more reliable labeler than SigLIP.

Table 2: Metrics d used in DSD. q_θ is a density function, s_{lang} is a text description, p_φ is a classifier, and $\Delta\theta$ is angular displacement. FoG uses the foundation model augmented hybrid distance.

LSD	CSD	METRA	LGSD	DoDont	PSD	Ours
$\|\|s' - s\|\|$	$-\log q_\theta$	temp. dist	$\text{dist}(s_{lang}, s'_{lang})$	$p_\varphi(s, s')$	$\Delta\theta$	hybrid

6 Conclusion and Future Work

FoG aligns unsupervised skill discovery with a preference stated once in language, supplied zero-shot by a foundation model and applied as a multiplicative gate on the diversity reward. The gate is not a reward, since it cannot be maximized on its own (Appendix C), which is what keeps FoG a skill-discovery method. Lifting the WDM to trajectories, we showed the gate enters through the KR-dual critic and that the per-step objective lower-bounds the trajectory-level dependency (Theorem 3.1), collapsing to METRA for a constant gate. Empirically, FoG eliminates flipping, avoids hazards, discovers language-specified postures without demonstrations, and yields libraries that adapt faster and more safely downstream. The binary gate is robust to label noise but sensitive to phrasing. Since w scores single states, FoG cannot express preferences over sequences; scoring short windows (Sontakke et al., 2023) is a direct remedy. Combining intentions composes too restrictively (Appendix F.4), and stronger VLMs (Liu et al., 2024; Yao et al., 2024) should help on harder tasks.

AI USE STATEMENT

Generative AI is a component of the method: FoG queries ChatGPT or Claude once per state-based task to write the score function (Appendix G.7.1) and uses CLIP to label pixel observations (Section 3.4); every prompt and every generated function is reported verbatim in Appendices G.7.1 and G.7.2. Separately, we used a large language model to edit and restructure the prose of this paper; all research ideas, derivations, proofs, experiments, and analyses are our own, and every AI-assisted passage was reviewed by the authors for accuracy. We take responsibility for the final content of this work.

ETHICS STATEMENT

The human evaluation in Section 4.2 asked ten adult volunteers to compare pairs of short videos of simulated robots (Appendix G.5); no personal data was collected and participation carried no risk. All experiments use simulated environments. FoG is designed to make discovered skills safer by suppressing hazardous behaviors, but the same mechanism can steer discovery toward whatever a sentence describes, and foundation-model labels inherit the biases and errors of the underlying model (Section 4.4); we therefore regard the gate as a specification tool that complements, rather than replaces, safety validation of the resulting skills.

REPRODUCIBILITY STATEMENT

Source code is provided as supplementary material. Complete proofs of Theorem 3.1 and its supporting lemmas, with all assumptions stated, are in Appendix B. Environment details, episode lengths, and observation sizes are in Appendix G.1; baseline implementations in Appendix G.2; training budgets and skill dimensions in Table 3; the foundation models used, with exact checkpoints, in Appendix G.6; and all prompts and intention sentences in Appendices G.7.1 and G.7.2. Every reported number is a mean over three seeds with the dispersion stated in the corresponding caption.

REFERENCES

Ademi Adeniji, Amber Xie, Carmelo Sferrazza, Younggyo Seo, Stephen James, and Pieter Abbeel. Language reward modulation for pretraining reinforcement learning. *arXiv preprint arXiv:2308.12270*, 2023.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

Ishan Durugkar, Mauricio Tec, Scott Niekum, and Peter Stone. Adversarial intrinsic motivation for reinforcement learning. *Advances in Neural Information Processing Systems*, 34:8622–8636, 2021.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022.

Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. In *Conference on Robot Learning (CoRL)*, 2024.

Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.

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, 2018.

Kristian Hartikainen, Xinyang Geng, Tuomas Haarnoja, and Sergey Levine. Dynamical distance learning for semi-supervised and unsupervised skill discovery. *arXiv preprint arXiv:1907.08225*, 2019.

Maxence Hussonnois, Thommen George Karimpanal, and Santu Rana. Controlled diversity with preference: Towards learning a diverse set of desired skills. *arXiv preprint arXiv:2303.04592*, 2023.

Hyunseung Kim, Byungkun Lee, Hojoon Lee, Dongyoon Hwang, Donghu Kim, and Jaegul Choo. Do's and don'ts: Learning desirable skills with instruction videos. *arXiv preprint arXiv:2406.00324*, 2024.

Sunin Kim, Jaewoon Kwon, Taeyoon Lee, Younghyo Park, and Julien Perez. Safety-aware unsupervised skill discovery. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 894–900. IEEE, 2023.

Martin Klissarov, Pierluca D'Oro, Shagun Sodhani, Roberta Raileanu, Pierre-Luc Bacon, Pascal Vincent, Amy Zhang, and Mikael Henaff. Motif: Intrinsic motivation from artificial intelligence feedback. *arXiv preprint arXiv:2310.00166*, 2023.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

Michael Laskin, Hao Liu, Xue Bin Peng, Denis Yarats, Aravind Rajeswaran, and Pieter Abbeel. Unsupervised reinforcement learning with contrastive intrinsic control. *Advances in Neural Information Processing Systems*, 35:34478–34491, 2022.

Kimin Lee, Laura Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26286–26296. IEEE, 2024.

Cong Lu, Shengran Hu, and Jeff Clune. Intelligent go-explore: Standing on the shoulders of giant foundation models. *arXiv preprint arXiv:2405.15143*, 2024.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

Edward James McShane. Extension of range of functions. 1934.

Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhai Wang, Mingyu Ding, Jun Jin, Bin Wang, Jifeng Dai, Yu Qiao, and Ping Luo. Embodiedgpt: Vision-language pre-training via embodied chain of thought. *Advances in neural information processing systems*, 36:25081–25094, 2023.

Taewook Nam, Juyong Lee, Jesse Zhang, Sung Ju Hwang, Joseph J Lim, and Karl Pertsch. Lift: Unsupervised reinforcement learning with foundation models as teachers. *arXiv preprint arXiv:2312.08958*, 2023.

Sherjil Ozair, Corey Lynch, Yoshua Bengio, Aaron Van den Oord, Sergey Levine, and Pierre Sermanet. Wasserstein dependency measure for representation learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Jonghae Park, Daesol Cho, Jusuk Lee, Dongseok Shim, Inkyu Jang, and H Jin Kim. Periodic skill discovery. *arXiv preprint arXiv:2511.03187*, 2025.

Seohong Park, Jongwook Choi, Jaekyeom Kim, Honglak Lee, and Gunhee Kim. Lipschitz-constrained unsupervised skill discovery. In *International Conference on Learning Representations*, 2022.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. Hiql: Offline goal-conditioned rl with latent states as actions. *Advances in Neural Information Processing Systems*, 36:34866–34891, 2023a.

Seohong Park, Kimin Lee, Youngwoon Lee, and Pieter Abbeel. Controllability-aware unsupervised skill discovery. *arXiv preprint arXiv:2302.05103*, 2023b.

Seohong Park, Oleh Rybkin, and Sergey Levine. Metra: Scalable unsupervised rl with metric-aware abstraction. *arXiv preprint arXiv:2310.08887*, 2023c.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Seungeun Rho, Laura Smith, Tianyu Li, Sergey Levine, Xue Bin Peng, and Sehoon Ha. Language guided skill discovery. *arXiv preprint arXiv:2406.06615*, 2024.

Juan Rocamonde, Victoriano Montesinos, Elvis Nava, Ethan Perez, and David Lindner. Vision-language models are zero-shot reward models for reinforcement learning. *arXiv preprint arXiv:2310.12921*, 2023.

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

Bharat Singh, Rajesh Kumar, and Vinay Pratap Singh. Reinforcement learning in robotic applications: a comprehensive survey. *Artificial intelligence review*, 55(2):945–990, 2022.

Arpita Soni, Sujatha Alla, Suresh Dodda, and Hemanth Volikatla. Advancing household robotics: Deep interactive reinforcement learning for efficient training and enhanced performance. *arXiv preprint arXiv:2405.18687*, 2024.

Sumedh Sontakke, Jesse Zhang, Séb Arnold, Karl Pertsch, Erdem Bıyık, Dorsa Sadigh, Chelsea Finn, and Laurent Itti. Roboclip: One demonstration is enough to learn robot policies. *Advances in Neural Information Processing Systems*, 36:55681–55693, 2023.

Henry Sowerby, Zhiyuan Zhou, and Michael L Littman. Designing rewards for fast learning. *arXiv preprint arXiv:2205.15400*, 2022.

Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. Deep reinforcement learning for robotics: A survey of real-world successes. *arXiv preprint arXiv:2408.03539*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638.

Miguel Vasco, Takuma Seno, Kenta Kawamoto, Kaushik Subramanian, Peter R Wurman, and Peter Stone. A super-human vision-based reinforcement learning agent for autonomous racing in gran turismo. *arXiv preprint arXiv:2406.12563*, 2024.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, Qianyu Chen, Huarong Zhou, Zhensheng Zou, Haoye Zhang, Shengding Hu, Zhi Zheng, Jie Zhou, Jie Cai, Xu Han, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint 2408.01800*, 2024.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11941–11952. IEEE, 2023.

Appendix

Table of Contents

A Impact Statement **14**

B Proofs **14**
B.1 Supporting lemmas 15
B.2 Proof of Theorem 3.1 16

C Relation to Demonstration- and Reward-Based Alternatives **18**

D Extended Related Work **18**

E Analysis of DoDont **19**

F Additional Experiments **20**
F.1 Quadruped Learns to Not Flip 20
F.2 Results on Franka Kitchen 20
F.3 Further Gate Ablations 20
F.4 Results on Multiple Intentions 21
F.5 Using scores as step-wise reward signals in FR-SAC 21

G Experiment Details **22**
G.1 Environment Details 22
G.2 Baseline Details 22
G.3 Non-binary Score Function 23
G.4 Computation Usage 24
G.5 Experimental Setup for Human Judge 24
G.6 Foundation Models 24
G.7 Prompts Used 25

A Impact Statement

Integrating foundation models to guide RL agents significantly enhances agent autonomy, yet simultaneously increases the risk of 'reward hacking', where agents pursue unintended strategies to maximize performance. This evolution necessitates a heightened focus on value alignment, interpretability, and safety to ensure that increasingly autonomous RL systems remain robust and beneficial

B Proofs

This appendix collects two supporting lemmas and the full proof of Theorem 3.1. Lemma B.1 controls the Lipschitz constant of the gated trajectory embedding under the hybrid metric, and Lemma B.2 converts

a global Lipschitz constraint on ϕ into an equivalent local one over adjacent state pairs. The proof of Theorem 3.1 chains these with KR duality and a centered-prior argument.

B.1 SUPPORTING LEMMAS

Lemma B.1 (Lipschitz continuity of Φ_ψ under the hybrid trajectory metric)**.** *Assume:*

(i) $\phi : \mathcal{S} \to \mathbb{R}^d$ *is* L_ϕ*-Lipschitz with respect to* d_{temp}*, i.e.,* $\|\phi(s) - \phi(s')\|_2 \le L_\phi\, d_{\text{temp}}(s, s')$ *for all* s, s'*;*

(ii) $\|\phi(s)\|_2 \le B_\phi$ *for all* s*;*

(iii) $w(s) \in \{\alpha, 1\}$ *with* $0 < \alpha \le 1$*, and let* $c : \mathcal{S} \to \{0, 1\}$ *be such that* $w(s) = \alpha$ *iff* $c(s) = 0$ *and* $w(s) = 1$ *iff* $c(s) = 1$*.*

Define the hybrid state distance

$$d_{\tilde{s}}(s, s') := d_{\text{temp}}(s, s') + \beta\, \mathbf{1}\{c(s) \neq c(s')\}, \qquad \beta > 0,$$

and the induced trajectory distance for equal-length trajectories

$$d_{\tilde{\tau}}(\tau, \tau') := \sum_{t=0}^{T} d_{\tilde{s}}(s_t, s'_t).$$

Then the gated trajectory embedding

$$\Phi_\psi(\tau) = \sum_{t=0}^{T-1} w(s_{t+1}) \big(\phi(s_{t+1}) - \phi(s_t)\big)$$

is Lipschitz with respect to $d_{\tilde{\tau}}$ *with constant*

$$L_\Phi \;=\; 2L_\phi \;+\; \frac{2B_\phi(1-\alpha)}{\beta}.$$

Proof. Write $\phi_t = \phi(s_t)$, $\phi'_t = \phi(s'_t)$, $w_{t+1} = w(s_{t+1})$, and $w'_{t+1} = w(s'_{t+1})$. Let $A_t = w_{t+1}(\phi_{t+1} - \phi_t)$ and $A'_t = w'_{t+1}(\phi'_{t+1} - \phi'_t)$. Then

$$\Phi_\psi(\tau) - \Phi_\psi(\tau') = \sum_{t=0}^{T-1} (A_t - A'_t), \qquad \|\Phi_\psi(\tau) - \Phi_\psi(\tau')\|_2 \;\le\; \sum_{t=0}^{T-1} \|A_t - A'_t\|_2.$$

Fix t and add and subtract $w_{t+1}(\phi'_{t+1} - \phi'_t)$:

$$A_t - A'_t = w_{t+1}\big[(\phi_{t+1} - \phi'_{t+1}) - (\phi_t - \phi'_t)\big] + (w_{t+1} - w'_{t+1})(\phi'_{t+1} - \phi'_t).$$

Taking norms and using $w_{t+1} \le 1$,

$$\|A_t - A'_t\|_2 \;\le\; \|\phi_{t+1} - \phi'_{t+1}\|_2 + \|\phi_t - \phi'_t\|_2 + |w_{t+1} - w'_{t+1}|\, \|\phi'_{t+1} - \phi'_t\|_2. \tag{11}$$

Representation-difference terms. By assumption (i) and $d_{\tilde{s}} \ge d_{\text{temp}}$,

$$\|\phi_{t+1} - \phi'_{t+1}\|_2 \le L_\phi\, d_{\tilde{s}}(s_{t+1}, s'_{t+1}), \qquad \|\phi_t - \phi'_t\|_2 \le L_\phi\, d_{\tilde{s}}(s_t, s'_t).$$

Gate-mismatch term. If $c(s_{t+1}) = c(s'_{t+1})$, then $w_{t+1} = w'_{t+1}$ and the term vanishes. Otherwise $|w_{t+1} - w'_{t+1}| = 1 - \alpha$ and $d_{\tilde{s}}(s_{t+1}, s'_{t+1}) \geq \beta$, so

$$|w_{t+1} - w'_{t+1}| \;\leq\; \frac{1-\alpha}{\beta}\, d_{\tilde{s}}(s_{t+1}, s'_{t+1})$$

in either case. Combined with $\|\phi'_{t+1} - \phi'_t\|_2 \leq \|\phi'_{t+1}\|_2 + \|\phi'_t\|_2 \leq 2B_\phi$ from (ii), this gives

$$|w_{t+1} - w'_{t+1}|\, \|\phi'_{t+1} - \phi'_t\|_2 \;\leq\; \frac{2B_\phi(1-\alpha)}{\beta}\, d_{\tilde{s}}(s_{t+1}, s'_{t+1}).$$

Substituting both estimates into equation 11,

$$\|A_t - A'_t\|_2 \;\leq\; L_\phi\big(d_{\tilde{s}}(s_{t+1}, s'_{t+1}) + d_{\tilde{s}}(s_t, s'_t)\big) + \frac{2B_\phi(1-\alpha)}{\beta}\, d_{\tilde{s}}(s_{t+1}, s'_{t+1}).$$

Summing over $t = 0, \ldots, T-1$ and using $\sum_{t=0}^{T-1}\big(d_{\tilde{s}}(s_{t+1}, s'_{t+1}) + d_{\tilde{s}}(s_t, s'_t)\big) \leq 2d_{\tilde{\tau}}(\tau, \tau')$ and $\sum_{t=0}^{T-1} d_{\tilde{s}}(s_{t+1}, s'_{t+1}) \leq d_{\tilde{\tau}}(\tau, \tau')$,

$$\|\Phi_\psi(\tau) - \Phi_\psi(\tau')\|_2 \;\leq\; \Big(2L_\phi + \frac{2B_\phi(1-\alpha)}{\beta}\Big)\, d_{\tilde{\tau}}(\tau, \tau'). \qquad \square$$

Lemma B.2 (Local Lipschitz constraint for d_{temp})**.** *Let $\mathcal{S}_{\text{adj}}$ denote adjacent state pairs (one environment step), and assume $d_{\text{temp}}(s, s') = 1$ for all $(s, s') \in \mathcal{S}_{\text{adj}}$. Define $d_{\text{temp}}(u, v)$ for general u, v as the temporal shortest-path distance induced by $\mathcal{S}_{\text{adj}}$. Then the following are equivalent:*

(a) $\|\phi(u) - \phi(v)\|_2 \leq d_{\text{temp}}(u, v)$ *for all* $u, v \in \mathcal{S}$*;*

(b) $\|\phi(s) - \phi(s')\|_2 \leq 1$ *for all* $(s, s') \in \mathcal{S}_{\text{adj}}$.

Proof. **(a)⇒(b).** For $(s, s') \in \mathcal{S}_{\text{adj}}$ we have $d_{\text{temp}}(s, s') = 1$, so (a) gives $\|\phi(s) - \phi(s')\|_2 \leq 1$.

(b)⇒(a). Fix u, v with $d_{\text{temp}}(u, v) = k < \infty$. By definition of d_{temp}, there exists a path $u = s_0, \ldots, s_k = v$ with $(s_i, s_{i+1}) \in \mathcal{S}_{\text{adj}}$ for each i. By the triangle inequality and (b),

$$\|\phi(u) - \phi(v)\|_2 \;\leq\; \sum_{i=0}^{k-1} \|\phi(s_i) - \phi(s_{i+1})\|_2 \;\leq\; k \;=\; d_{\text{temp}}(u, v).$$

If $d_{\text{temp}}(u, v) = \infty$, the inequality is trivial. $\square$

Remark. Since $d_{\tilde{s}} \geq d_{\text{temp}}$, condition (b) also implies $\|\phi(u) - \phi(v)\|_2 \leq d_{\tilde{s}}(u, v)$ for all u, v, so the same local constraint controls Lipschitzness under the hybrid metric.

B.2 Proof of Theorem 3.1

We derive the inequality

$$W_1^{(d_{(\tau,z)})}\big(P_\pi(\tau, z),\, P_\pi(\tau)\, p(z)\big) \;\geq\; \frac{1}{L_\Phi}\, \mathbb{E}_{z\sim p(z),\, \tau\sim P_\pi(\cdot|z)}\left[\sum_{t=0}^{T-1} w(s_{t+1})\big(\phi(s_{t+1}) - \phi(s_t)\big)^\top z\right] \tag{12}$$

in four steps.

Step 1: Local-to-global Lipschitzness of ϕ. By assumption, ϕ satisfies $\|\phi(s) - \phi(s')\|_2 \le 1$ for all $(s, s') \in \mathcal{S}_{\text{adj}}$. Lemma B.2 then gives $\|\phi(u) - \phi(v)\|_2 \le d_{\text{temp}}(u, v)$ for all $u, v \in \mathcal{S}$, i.e., ϕ is 1-Lipschitz with respect to d_{temp} ($L_\phi = 1$).

Step 2: Lipschitzness of the gated trajectory embedding. With $\|\phi(s)\|_2 \le B_\phi$ and $w(s) \in \{\alpha, 1\}$, the hypotheses of Lemma B.1 hold with $L_\phi = 1$, yielding

$$\|\Phi_\psi(\tau) - \Phi_\psi(\tau')\|_2 \;\le\; L_\Phi \, d_{\tilde{\tau}}(\tau, \tau'), \qquad L_\Phi \;=\; 2 + \frac{2B_\phi(1-\alpha)}{\beta}.$$

Moreover, since ϕ is 1-Lipschitz under d_{temp} (Step 1) and trajectory transitions are adjacent ($d_{\text{temp}}(s_t, s_{t+1}) = 1$), each summand satisfies $\|\phi(s_{t+1}) - \phi(s_t)\|_2 \le 1$. Combined with $|w(s_{t+1})| \le 1$,

$$\|\Phi_\psi(\tau)\|_2 \;\le\; \sum_{t=0}^{T-1} |w(s_{t+1})| \, \|\phi(s_{t+1}) - \phi(s_t)\|_2 \;\le\; T.$$

We denote this bound by $B_\Phi := T$.

Step 3: A 1-Lipschitz normalized critic. Define the candidate critic on the support of $p(z)$:

$$\tilde{f}_\psi(\tau, z) := \frac{1}{L_\Phi} \langle \Phi_\psi(\tau), \, z \rangle, \qquad (\tau, z) \in \mathcal{T} \times \{\|z\|_2 \le 1\}.$$

For $(\tau, z), (\tau', z')$ on this domain, add and subtract $\langle \Phi_\psi(\tau'), z \rangle$:

$$L_\Phi\big(\tilde{f}_\psi(\tau, z) - \tilde{f}_\psi(\tau', z')\big) = \langle \Phi_\psi(\tau) - \Phi_\psi(\tau'), \, z \rangle + \langle \Phi_\psi(\tau'), \, z - z' \rangle.$$

By Cauchy-Schwarz, Step 2, and $\|z\|_2 \le 1$,

$$|\tilde{f}_\psi(\tau, z) - \tilde{f}_\psi(\tau', z')| \;\le\; \frac{1}{L_\Phi}\big(L_\Phi \, d_{\tilde{\tau}}(\tau, \tau') + B_\Phi \, \|z - z'\|_2\big) = d_{\tilde{\tau}}(\tau, \tau') + \frac{T}{L_\Phi} \, \|z - z'\|_2.$$

With $\lambda = T/L_\Phi$, this says $\tilde{f}_\psi$ is 1-Lipschitz with respect to $d_{(\tau,z)}$ on $\mathcal{T} \times \{\|z\|_2 \le 1\}$. By McShane's extension theorem (McShane, 1934), there exists $\bar{f}_\psi : \mathcal{T} \times \mathcal{Z} \to \mathbb{R}$ that is 1-Lipschitz with respect to $d_{(\tau,z)}$ on the whole space and agrees with $\tilde{f}_\psi$ on $\mathcal{T} \times \{\|z\|_2 \le 1\}$. Since both $P_\pi(\tau, z)$ and $P_\pi(\tau)p(z)$ are supported on this set (the prior $p(z)$ being supported on $\{\|z\|_2 \le 1\}$ by hypothesis), $\bar{f}_\psi$ is a valid 1-Lipschitz competitor in the KR dual whose expectations under both measures equal those of $\tilde{f}_\psi$.

Step 4: KR duality and the centered prior. By KR duality applied to $W_1^{(d_{(\tau,z)})}$,

$$W_1^{(d_{(\tau,z)})}\big(P_\pi(\tau, z), \, P_\pi(\tau)p(z)\big) \;=\; \sup_{\|f\|_{\text{Lip}} \le 1} \Big\{ \mathbb{E}_{P_\pi(\tau,z)}[f] - \mathbb{E}_{P_\pi(\tau)p(z)}[f] \Big\}.$$

Plugging in the 1-Lipschitz extension $\bar{f}_\psi$ from Step 3 and using $\mathbb{E}[\bar{f}_\psi] = \mathbb{E}[\tilde{f}_\psi]$ under both measures,

$$W_1^{(d_{(\tau,z)})}(\cdots) \;\ge\; \mathbb{E}_{P_\pi(\tau,z)}\big[\tilde{f}_\psi(\tau, z)\big] - \mathbb{E}_{P_\pi(\tau)p(z)}\big[\tilde{f}_\psi(\tau, z)\big].$$

The second term factorizes through the inner product:

$$\mathbb{E}_{P_\pi(\tau)p(z)}\big[\tilde{f}_\psi(\tau, z)\big] = \frac{1}{L_\Phi} \, \mathbb{E}_{P_\pi(\tau)}\big[\langle \Phi_\psi(\tau), \, \mathbb{E}_{p(z)}[z] \rangle\big] = 0,$$

since $\mathbb{E}_{p(z)}[z] = 0$. Substituting and expanding Φ_ψ gives

$$W_1^{(d_{(\tau,z)})}(\cdots) \;\ge\; \frac{1}{L_\Phi} \, \mathbb{E}_{z \sim p(z), \, \tau \sim P_\pi(\cdot|z)} \left[\sum_{t=0}^{T-1} w(s_{t+1}) \big(\phi(s_{t+1}) - \phi(s_t)\big)^\top z \right].$$

The constrained-supremum statement of Theorem 3.1 is immediate: the right-hand side of equation 12 is a deterministic function of (π, ϕ) on the constraint set $\{\|\phi(s) - \phi(s')\|_2 \leq 1 \text{ on } \mathcal{S}_{\text{adj}}\} \times \{\|z\|_2 \leq 1\}$, while the left-hand side is a property of π alone. Maximizing the right-hand side over (π, ϕ) subject to these constraints therefore maximizes a constant-factor lower bound on $W_1^{(d_{(\tau,z)})}\big(P_\pi(\tau, z),\ P_\pi(\tau)p(z)\big)$. □

C Relation to Demonstration- and Reward-Based Alternatives

Every method we compare against in Section 4 can be written as a modification of the per-step reward in equation 9, which makes the differences exact. Write $r_t^{\text{div}} := (\phi(s_{t+1}) - \phi(s_t))^\top z$ for METRA's diversity reward. FoG trains on $w(s_{t+1})\, r_t^{\text{div}}$ with w binary, defined on the state entered, and supplied zero-shot by a foundation model.

DoDont (Kim et al., 2024) trains on $p_\varphi(s_t, s_{t+1})\, r_t^{\text{div}}$, where p_φ is a classifier trained on "do" and "don't" demonstration videos. The gate sits in the same multiplicative position, so at the level of the reward both methods are gated METRA. They differ in three respects. First, in what supplies the gate: a classifier that must be trained from demonstrations the user has to collect, versus a label obtained from a sentence with no data. This determines which preferences can be expressed at all (there are no demonstration videos of "twisted") and the failure mode (Appendix E shows the classifier learning a background-color shortcut). Second, in where the gate is defined: on the transition pair (s_t, s_{t+1}) versus on the state entered, s_{t+1}, which Section 3.2 motivates as the natural domain of a preference over outcomes. Third, in derivation: DoDont introduces the gate as a choice of distance in the DSD framework; the trajectory-level derivation of Section 3 shows the gated reward lower-bounds a dependency measure and explains why the state-level derivation cannot host it.

METRA+ trains on $r^{\text{hc}}(s_{t+1})\, r_t^{\text{div}}$ with a hand-coded score r^{hc} (Kim et al., 2024). When the preference has an obvious hand-coded form, such as an indicator of the safe region, METRA+ is an oracle for FoG's gate and should upper-bound what FoG can achieve; Section 4.2 reports exactly this on the area-avoidance tasks. FoG is aimed at the complementary regime, where the hand-coded form is unclear (the $r_{run} - r_{flip}$ score used for flipping underperforms) or nonexistent ("twisted").

LGSD (Rho et al., 2024) keeps r_t^{div} ungated and instead changes the constraint in equation 9 to $\|\phi(s') - \phi(s)\|^2 \leq d_{\text{lang}}(s, s')$, where d_{lang} is a distance between language descriptions of the two states that carry the user's intent (Appendix G.2). Intent therefore enters as a metric rather than a gate: undesirable states are made close in representation space, so no skill is rewarded for reaching them, but nothing prevents a skill from passing through them, and in Figure 3 LGSD keeps METRA's coverage while still flipping in a quarter of episodes. FoG's hybrid metric equation 8 also carries the preference, but the gate on the reward is what removes the undesirable transitions.

METRA+FR and FR-SAC train on $r_t^{\text{div}} + w(s_{t+1})$ and on $w(s_{t+1})$ alone, respectively. They isolate a natural question: since w induces a preference ordering over states, why is FoG not simply RL with w as the reward? The answer is that a reward is something the agent maximizes, and the gate is not. w is bounded by 1 and attains it in any preferred state, including the initial one, so the behavior that maximizes w on its own is to stand still, which is precisely what FR-SAC and METRA+FR learn (Figure 3). In FoG the gate multiplies the diversity reward, so it can only scale the credit for moving, never substitute for it. No single behavior is optimal for equation 9; the optimum is a family of skills indexed by z, all of which move, restricted to the preferred region. That is what keeps FoG a skill-discovery method rather than a reward-following one.

D Extended Related Work

Foundation Models in Reinforcement Learning FoG leverages foundation models to guide unsupervised skill discovery in learning desirable behaviors. Thanks to success of foundation models (Touvron et al., 2023; Liu et al., 2023) they can now be used to provide information for RL agents. Motif (Klissarov et al., 2023) and IGE (Lu et al., 2024) employs large language models to generate exploration bonuses. Eureka (Ma et al., 2023) uses large language models to generate reward functions for state-based robotic tasks, outperforming human designed reward functions across multiple tasks. Lift (Nam et al., 2023) uses LLM and VLM to guide learning in Minedojo (Fan et al., 2022). LAMP (Adeniji et al., 2023) and Rocamonde et al. (2023) utilize the similarity between pixel embedding and text-commands embedding, as output by a vision-language model, as the step-wise reward in visual robotic tasks. Results show that such step-wise signals alone barely work (matching the results we had in Section 4.4), and require either fine-tuning or special task modifications to perform well. Task-specific foundation models generally can achieve better performance on specific tasks, such as Minedojo (Fan et al., 2022) in Minecraft and EmbodiedGPT (Mu et al., 2023) in robotics. Despite this, FoG demonstrates that pre-trained foundation models, even without fine-tuning or any modifications of tasks, can be used to guide RL agents to discover diverse and desirable skills. In state-based tasks, FoG uses foundation models to generate a score function aligned with human intentions. Unlike Eureka (Ma et al., 2023), FoG: 1) avoids iterative feedback loops with the environment, as Eureka requires multiple rounds of feedback to refine the reward function, and 2) uses the score function to re-weight skill discovery rewards, whereas Eureka directly trains agents with the generated reward function.

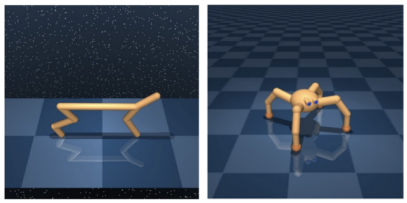

Figure 7: Tasks with non-colored ground that DoDont uses.

E Analysis of DoDont

The performance of DoDont in our paper is quite different to the one from the original paper due to different experimental setup. Here, we provide a more in-depth analysis of why DoDont fails in our experiments.

Failure in Figure 3 To keep a fair comparison, we use pixel inputs for both the classifier and the RL part in DoDont (since FoG does not require ground-truth state information and works with only pixel inputs), which differs from the original experiments in the DoDont paper. In the Appendix D.1 of DoDont paper, authors mentioned that DoDont uses state information as input for the RL agent (both the policy and the critic). The classifier might exploit the background color as a shortcut to distinguish between different states rather than observing the agent's embodiment, thus DoDont instead uses a non-colored ground (see in Figure 7). However, the backbone of DoDont, METRA, cannot learn diverse skills without the colored ground (since there will be no indication of directions). Thus, DoDont uses state-information for the RL part. During the training, image states are first input to the classifier to get rewards, then the corresponding compact ground-truth states are used to train the RL agents along with the rewards obtained from the classifier. Our experiments show that indeed, if pixel inputs are used for both the RL part and the classifier, DoDont fails (see results in Figure 3). The classifier indeed exploits the background color as a shortcut to distinguish between different states rather than observing the agent's embodiment, classifies unseen 'Dont' states as 'Do'. See videos on `https://sites.google.com/view/submission-fog`.

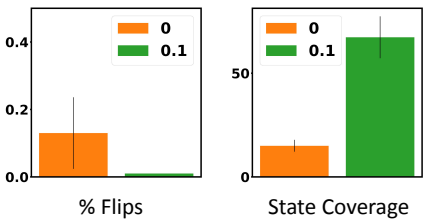

Figure 8: Results on the Quadruped task. Setting $\alpha = 0$ explores less (lower state coverage) thus results in worse performance (more flips).

Behavior in Figure 4 In the area-avoidance tasks DoDont's demonstrations are labeled with the ground-truth region indicator, and with that privileged supervision its classifier separates "safe" from "hazardous" frames reliably, so DoDont attains high safe-state coverage in both Cheetah and Quadruped. The failure mode reappears as soon as the labels come from the foundation model instead: DoDont+ trains the same classifier on CLIP-annotated demonstrations of about 70% accuracy and covers little of the safe region, especially in Cheetah. See videos on `https://sites.google.com/view/submission-fog`.

F Additional Experiments

F.1 Quadruped Learns to Not Flip

Although we found that setting $\alpha = 0$ works well in experiments presented in Section Experiments, sometimes it might hurt the exploration. Similar with experiments performed in Figure 3, here, we train FoG to not flip in Quadruped. We see in Figure 8, FoG learns to not flip most of time (less than 20%) when setting $\alpha = 0$, but it almost always stays near the starting point and does not explore, resulting in low state coverage. After loosing α a bit and set it to 0.1, FoG learns to eliminate all flips and has a significant higher state coverage.

F.2 Results on Franka Kitchen

To examine FoG in more complicated tasks, we train FoG in Franka Kitchen (introduced by Gupta et al. (2019)) with different textual descriptions of intentions, such as `'robotic arm is stretched'`, `'robotic arm is twisted'` and `'robotic arm is on the right of the scene'`. Results can be seen in Figure 9. By using different intentions, we see robotic arms clearly bias the movements to different areas. However, we did not find a way to use these skills to better solve the downstream tasks yet. We hope this could inspire future efforts in investigating FoG in more complex tasks.

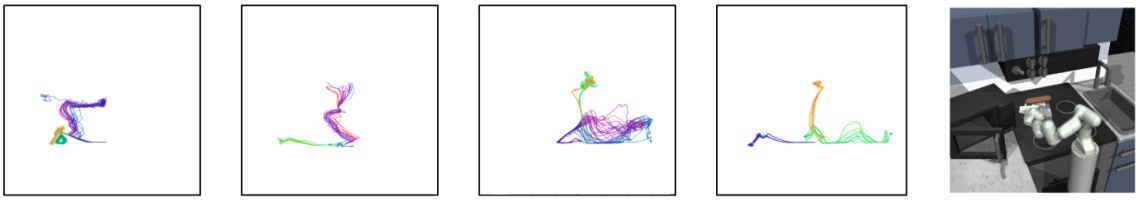

Figure 9: In Franka Kitchen, different skills FoG learned with different textual descriptions of intentions. Skills are displayed with x-y coordinates of the robotic arm.

We also found that the baseline method METRA learns to open the cabinet very often, e.g. in our case 78% of time. We test FoG to bias towards `'cabinet is closed'`. It learns to stop opening the cabinet (only 22%), and please see the videos on our project website. We see the agent stops the intention to open the cabinet in most of cases.

F.3 Further Gate Ablations

Figure 10 reports the two gate ablations not shown in Figure 6: the labeling stride N and the phrasing of the intention sentence. Both are discussed in Section 4.4.

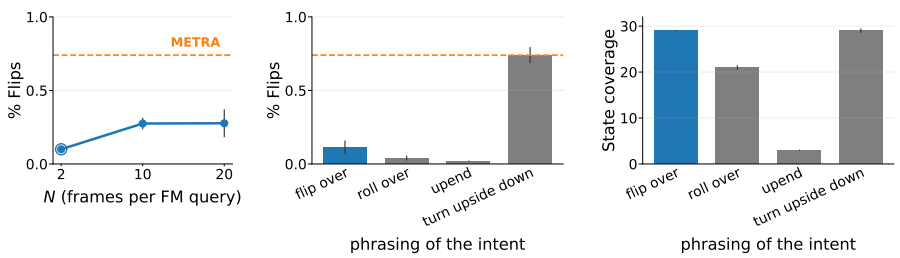

Figure 10: Further gate ablations on Cheetah (fraction of flipped episodes; dashed line: METRA). Left: querying the foundation model every N frames; N=2 (default, hollow marker) is safest and larger N costs about 0.2. Right: phrasing of the same intent; "flip over" (default, blue) keeps METRA's coverage, "turn upside down" does not suppress flipping, and "roll over"/"upend" avoid flips only by moving less (coverage 21 and 3). Mean ± standard error.

F.4 Results on Multiple Intentions

In Section Experiments, only one intention is used in FoG. In principle, multiple intentions could be used simultaneously to form the score function. Then, Equation (10) becomes:

$$w(s) = \begin{cases} 1, & \text{if } Cosine(E_s, E^1_{t1}) > Cosine(E_s, E^1_{t2}) \text{ and} \\ & \quad Cosine(E_s, E^2_{t1}) > Cosine(E_s, E^2_{t2}) \text{ and} \\ & \quad \ldots \\ & \quad Cosine(E_s, E^n_{t1}) > Cosine(E_s, E^n_{t2}) \\ \alpha, & \text{otherwise.} \end{cases} \tag{13}$$

where E^n_{t1} and E^n_{t2} are the nth textual descriptions of our intentions. Now, the score function $w(s)$ only assigns higher values to desirable states when all provided intentions are satisfied. For example, we could ask FoG to not only learns to not flip, but also to avoid the right area. The textual descriptions we should use are: 1) `'agent flips over'`, `'agent stands normally'`; 2) `'ground is Yellow-Orange'`, `'ground is Green-Blue'`. See the result in Figure 11, the agent does not learn to avoid the right part at all but it does learn to eliminate flips (not shown in the figure). We found that using multiple intentions restricts the exploration too much so that the agent might just learn to fulfill one intention and ignore others or ignore all of them and learns to not move at all. Using multiple intentions in FoG still needs more investigations and we hope the preliminary results and ideas presented in this section could inspire future efforts.

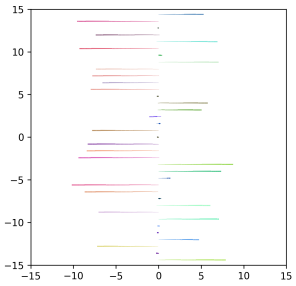

Figure 11: Skills learned by FoG with two intentions.

F.5 Using scores as step-wise reward signals in FR-SAC

FoG uses foundation model scores to re-weight the unsupervised skill discovery rewards, learning diverse and desirable behaviors. However, directly optimizing these scores is not ideal. In Figure 12, scores for pre-collected episodes aligned with human intentions ('Yes') and misaligned ones ('No') reveal significant noise despite correct overall trends (we use the same textual intentions from previous experiments, i.e. Cheetah in Figure 3 and Quadruped in Figure 4). For example, in Cheetah, after flipping upside down at step 50, the agent consistently receives low scores. In Quadruped, scores either remain high or gradually

decrease as the agent moves diagonally. This noise makes direct score optimization unreliable. As can be seen in Figure 3 and Figure 4, the agent trained solely with such noisy reward signals (FR-SAC) learns only static postures, resulting in low (safe) state coverages, suggesting that directly optimizing these scores is insufficient.

G Experiment Details

G.1 Environment Details

State-based HalfCheetah and Ant are from OpenAI Gym (Brockman et al., 2016). The state space of HalfCheetah is 18-dimensional and the one of Ant is 29-dimensional. HalfCheetah has a 6-dimensional action space while Ant has a 8-dimensional action space.

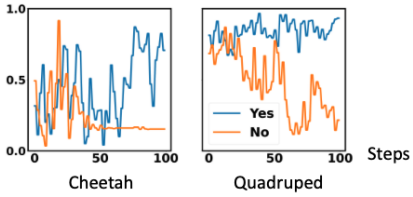

Figure 12: Scores outputted by foundation models on the pre-collected episode that are (not) aligned with human intentions.

Pixel-based Cheetah, Quadruped and Humanoid are from DeepMind Control Suite (Tunyasuvunakool et al., 2020). Following previous work (Lee et al., 2021; Park et al., 2023a;c), pixel-based DMC tasks are all with gradient-colored floors to indicate different directions. The size of visual observations is $64 \times 64 \times 3$. The dimension of action space for Cheetah, Quadruped and Humanoid are 6, 12 and 21, respectively. The episode length is 200 for Ant, HalfCheetah and Cheetah, 400 for Quadruped and Humanoid.

Modified Humanoid Since none of existing unsupervised skill discovery methods can train the visual Humanoid agent to stand up, limiting FoG to showcase more interesting behaviors, such as running, etc. We created a 'Puppet' task based on the DMC Humanoid environment, see Figure 13. The humanoid robot is pulled by a puppet anchor on the top of its head. Thus, the humanoid robot keeps standing by default and never falls down. The anchor also moves with the humanoid.

G.2 Baseline Details

METRA We take the official codebase[2] from Park et al. (2023c) and use default hyperparameters for all experiments performed in this paper.

METRA+ We follow the implementation of METRA+ in the DoDont paper. For experiments in Figure 3, we use $r_{run} - r_{flip}$ as the reward. For experiments in Figure 4, we assign +1 for the safe region and 0 for the hazardous region.

LSD We take the codebase of METRA, by setting correct arguments (turning off the dual regularization and turning on the spectral normalization), to run LSD. Detailed instructions can be found in the METRA codebase.

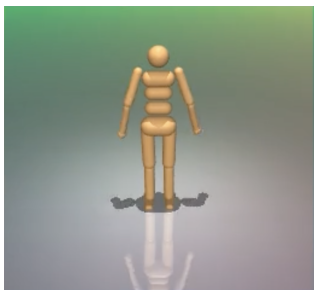

Figure 13: The Puppet environment.

DoDont We take the official codebase from Kim et al. (2024) and implement the training of the instruction net ourselves. We use eight demonstrations for each task, so four for "dos" and four for

[2] https://github.com/seohongpark/METRA

"donts". Demonstrations are obtained from trained FoG agents and can be found on our project website: `https://sites.google.com/view/submission-fog`. We stop the training of the classifier after it has more than 97% of accuracy.

DoDont+ A variant of DoDont, instead of using expert-level demonstrations, it uses demonstrations annotated by foundation models. In our case, we use CLIP to score frames (follow Equation (10)) in demonstrations that are used to train DoDont, and assign frames with score of 0 in the "dos" demonstration to "donts" demonstrations, and vice versa. Since CLIP cannot perfectly score frames, some states from "dos" demonstration are moved to "donts" demonstrations, and some states from "donts" demonstration are moved to "dos" demonstration. After training, the classifier of DoDont+ has about 70% of accuracy.

FR-SAC A soft actor-critic RL agent with using the score function as the reward function. We reuse the FoG codebase and set the number of skills to 1. Then, we train the skill-conditioned policy with the scores obtained from the foundation model (i.e. using the score function as the reward function), reducing to a normal RL agent.

LGSD Language-Guided Skill Discovery Rho et al. (2024) replaces METRA's temporal-distance constraint by a language distance. We implement it on top of the METRA codebase: the constraint $\|\phi(s') - \phi(s)\| \leq 1$ becomes $\|\phi(s') - \phi(s)\|^2 \leq d_{\text{lang}}(s, s')$, where d_{lang} is the distance between sentence embeddings (all-MiniLM-L6-v2) of natural-language descriptions of s and s'. Descriptions are produced by an LLM (Qwen2.5-7B-Instruct) from a rule-based textual annotation of the privileged simulator state (position, whether the agent is flipped, colour of the floor under it), using a prompt that carries the user's intent, e.g. "if the robot is upright, give its x position; if it has flipped over, only say so". Descriptions are cached per annotation, so the LLM is queried once per distinct state description. Unlike FoG, LGSD therefore consumes privileged state rather than pixels, and it encodes the intent as a distance, not as a gate: it can only make undesirable states close in skill space, not remove them. All other hyper-parameters are those of METRA.

METRA+FR A variant that adds the foundation-model score to METRA's reward instead of multiplying it: $r = (\phi(s') - \phi(s))^\top z + \lambda\, p_{\text{FM}}(s')$, where $p_{\text{FM}}(s') \in [0, 1]$ is CLIP's probability that the state matches the desirable description (the soft version of the score in Equation (10)) and λ=1. This is the natural "reward shaping" alternative to FoG's gate ("FR" for foundation reward). We reuse the FoG codebase with the same prompts and skill dimension. Because the added term is largest in states that are already judged desirable, it rewards staying where the score is high rather than exploring, and in all our tasks the agent becomes static near the start (coverage $\approx$ 2 cells), which is why METRA+FR appears with near-zero flips and near-zero coverage in Figures 3 and 4.

G.2.1 Hyperparameters Details

We use $\alpha = 0$ and $N = 2$ for all our experiments, unless otherwise mentioned. We train all agents in the same task with the same number of epochs and the performance at the end of training is reported. Details can be seen in Table 3. The same number of episodes is executed in each epoch, and within each episode the same number of environment steps is taken. We train continuous skills and the number of dimensions we used to train all agents in each task can be found in Table 3. We refer readers to read Park et al. (2023c) for details of all used hyperparameters.

G.3 Non-binary Score Function

Instead of using a binary score function in Equation (10), we can also form a non-binary score function.

Table 3: Number of epochs and dimensions of skills we used for training agents in different environments.

HalfCheetah	Ant	Cheetah	Quadruped	Humanoid
9000	9000	2000	3000	4000
4D	2D	4D	4D	2D

$$w(s) = \frac{e^{Cosine(E_s, E_{text1})}}{e^{Cosine(E_s, E_{text1})} + e^{Cosine(E_s, E_{text2})}}, \tag{14}$$

where E_s is the embedding of the current pixel-based state, E_{text1} is the embedding of textual descriptions of the desirable intention and E_{text2} is the embedding of textual descriptions of the undesirable intention.

G.4 Computation Usage

We run our experiments on an internal cluster consisting of A100 and H100 GPUs. Each run takes no more than 24 hours. Since FoG utilizes foundation models, it incurs additional computational overhead, approximately 1.5x slower than METRA.

G.5 Experimental Setup for Human Judge

In Figure 5, we train FoG to be twisted in DMC Humanoid task. However, it is difficult for human to design a reward function to measure if learned skills contains more twisted postures or not. Thus, we ask human to be the judge to tell if FoG learns more twisted skills than the ones learned by the baselines.

We pick ten skills of each method randomly, in this case, FoG and METRA, and then pair them randomly. Participates are asked to select the video that shows the most 'twisted' behaviors without given any other information. Please see the full questionnaire we used in `https://sites.google.com/view/submission-fog/questionnaire-of-humanoid`.

G.6 Foundation Models

For state-based tasks, we query ChatGPT[3] or Cluade[4] to generate score functions that meet our requirements. For pixel-based tasks, we use pre-trained CLIP (clip-vit-large-patch14) from huggingface[5].

[3] `https://chatgpt.com`

[4] `https://claude.ai/new`

[5] `https://huggingface.co/openai/clip-vit-large-patch14`

G.7 PROMPTS USED

G.7.1 STATE-BASED TASKS

Input to foundation models for HalfCheetah `[Descriptions of the task, state space and action space] According to the given info, could you please write a python function to check if the cheetah is flipped over or not. If yes, output 1 otherwise 0.` **Output:**

python Copy code

```python
import numpy as np

def is_cheetah_flipped(observation):
    """
    Check if the HalfCheetah is flipped over.

    Args:
    observation (np.array): Observation array from the environment.

    Returns:
    int: 1 if the cheetah is flipped over, 0 otherwise.
    """
    # The second element in the observation array is the angle of the front tip
    angle = observation[1]

    # Define the threshold for flipping (e.g., |angle| > π/2)
    threshold = np.pi / 2

    if abs(angle) > threshold:
        return 1
    else:
        return 0

# Example usage:
# Assume `obs` is the current observation from the environment
# obs = env.reset()
# print(is_cheetah_flipped(obs))
```

Input to foundation models for Ant `[Descriptions of the task, state space and action space] The agent always starts at [0, 0]. According to the given info, could you please write a python function to check if the agent is in the north part or not. If yes, output 1 otherwise 0.` **Output:**

python Copy code

```python
def is_in_north(state):
    """
    Check if the agent is in the northern part (y > 0).

    Parameters:
    state (list or numpy array): The state space observation vector. The y-posit

    Returns:
    int: 1 if the agent is in the northern part, 0 otherwise.
    """
    # Assuming the second element in the state space corresponds to the y-coordi
    y_position = state[1]

    # Check if the y-coordinate is greater than 0
    if y_position > 0:
        return 1
    else:
        return 0
```

G.7.2 PIXEL-BASED TASKS

Textual descriptions of intentions we used for Cheetah:

- Figure 3: `'The simulated two-leg robot flips over', 'The simulated two-leg robot stands normally'`
- FR-SAC agent in Figure 3: `'The simulated two-leg robot flips over', 'The simulated two-leg robot is running normally'`
- Figure 4: `'The underneath plane is Yellow-Orange', 'The underneath plane is Green-Blue'`

Textual descriptions of intentions we used for Quadruped in Figure 4: `'The underneath plane is Pink-Purple', 'The underneath plane is Green-Blue'`.

Textual descriptions of intentions we used for Humanoid in Figure 5: `'The simulated humanoid robot is stretched', 'The simulated humanoid robot is twisted'`.