Learning to Walk Like Humans Do:

A Developmental Approach to Locomotion in Deep Reinforcement Learning

Natalia Espinosa Dice

Advisor: Professor Tom Griffiths

Submitted in partial fulfillment

of the requirements for the degree of

Bachelor of Science in Engineering

Department of Computer Science

Princeton University

April 2026

I hereby declare that I am the sole author of this thesis.

I authorize Princeton University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Natalia Espinosa Dice ·

I further authorize Princeton University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Natalia Espinosa Dice ·

Abstract

Humans learn to stand and walk within a matter of months, exploiting a reliable sequence of postural milestones that partitions the full motor configuration space into a series of stable, tractable subproblems. In contrast, humanoid agents trained with standard reinforcement learning (RL) enjoy no such structure, confronting the full complexity of locomotion at once and facing an enormous exploration burden in high-dimensional continuous control. To address this, we propose a hierarchical reinforcement learning (HRL) framework that leverages the structure of human motor development by training each key postural transition—prone-to-crawl, crawl-to-kneel, kneel-to-lunge, lunge-to-stand and stand-to-walk—as a distinct low-level policy, thus imposing a developmental prior over the skill set. A high-level policy then learns to coordinate these motor skills via the Value Function Spaces (VFS) framework. We first validate our approach in LunarLander and BipedalWalker before applying it to Humanoid, where developmental structure directly informs the skill decomposition. Against flat RL baselines and HRL methods that discover structure from experience, our approach achieves upright posture in over 90% of evaluation episodes and completes the full developmental sequence in over 85%, while all baselines plateau at intermediate configurations. Our findings suggest that developmentally grounded structural priors substantially reduce the exploration burden of complex locomotion learning, enabling reliable postural progression where reward engineering, intrinsic motivation and emergent hierarchies fall short.

Acknowledgements

I am grateful to the mentors, collaborators, friends and family who made the last four years of learning so rich and rewarding, a journey that culminates in the work presented here.

I thank Gianluca Bencomo for his mentorship during the early stages of this project and Professor Brenden Lake for serving as a second reader.

I thank Professor Tom Griffiths for his guidance and feedback throughout this project and several draft iterations. When I joined his lab last year, I had limited formal research experience and little sense of what awaited me. I found an abundance of mentors willing to let me learn from them and a warmth of welcome that helped inspire my pursuit of research. This project has been the most influential learning experience of my undergraduate experience, and I am very grateful to have had the opportunity to pursue it under his advisorship.

Finally, I am deeply indebted to Dilip Arumugam for his infinite patience and support over the past year. I left every meeting with a greater sense of clarity and confidence than I entered, and I am grateful beyond words for his willingness to engage so thoughtfully with every detail of this project, from the first vague ideas to the final sprint. I could not have asked for a better mentor to help me navigate the week-to-week ups and downs of research, and this project truly would not have been possible without him.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Humans learn to walk with remarkable efficiency. Within eight to eighteen months, infants transition from complete immobility to confident bipedal locomotion [1]. In contrast, humanoid agents trained with reinforcement learning (RL) struggle with even the earliest foundations of movement. RL is a powerful framework for learning behavior through trial-and-error interaction with an environment and has achieved remarkable results across a range of motor control tasks [2, 3]. Yet, getting up from the ground, maintaining balance while standing and initiating forward motion often require orders of magnitude more experience than humans and still tend to result in inefficient or unnatural gaits [4, 5]. This disparity suggests that human motor development exploits structure in the learning problem that standard RL approaches fail to capture.

To understand the nature of this structure, we turn to developmental psychology, which reveals that infants do not approach walking as a monolithic control problem. Instead, they progress through a series of increasingly complex postures, which include lying prone, crawling, kneeling, half-kneeling, standing and finally walking [1, 6]. This progression follows a biomechanical logic in which each increasingly upright configuration narrows the base of support while raising the center of mass, such

that each stage introduces a distinct set of balance and coordination demands while restricting the degrees of freedom that must be simultaneously controlled [1, 7]. Together, these stages partition the vast configuration space of full body motor control into a sequence of simpler, more tractable subspaces.

This observation suggests a natural inductive bias for the learning problem. Agents trained with a generic RL formulation face the full complexity of locomotion at once and must learn balance, coordination and forward motion without guidance about which intermediate body configurations are useful. This makes exploration particularly challenging, as the agent must search through an extremely large space of possible configurations where most trajectories lead to falls or unstable states. In contrast, structuring learning around posture-specific transitions constrains exploration to regions of the motor space that are both stable and behaviorally meaningful. Our hypothesis is that by imposing a developmentally informed prior over which postural transitions are useful, we can substantially reduce the exploration burden and provide a coherent pathway toward full locomotion.

To instantiate this developmental decomposition of locomotion, we adopt a hierarchical reinforcement learning (HRL) framework in which low-level policies handle specific subtasks and a high-level policy coordinates between them [8]. This hierarchical structure separates short-horizon motor execution from long-horizon decision-making, allowing complex locomotion to emerge through the composition of simpler skills [9]. While most HRL methods require the agent to learn a skill decomposition from experience [10, 11, 12], we instead impose a set of skills informed directly by human motor development. Specifically, we first train each key postural transition in the developmental sequence—prone-to-crawl, crawl-to-kneel, kneel-to-lunge, lunge-to-stand and stand-to-walk—as a distinct motor skill with its own objective. This developmentally grounded structure provides a prior on which skills are useful, bypassing the difficult problem of skill discovery altogether.

To learn a high-level policy to coordinate these skills, we adopt the Value Function Spaces framework [13]. Rather than reasoning over the full motor state space, the high-level policy operates only over the value functions of the pretrained skills. Each skill's value function reflects the expected return under that skill from the current state, implicitly capturing whether the current configuration affords that transition. Stacking these value functions thus compresses the high-dimensional motor state into a low-dimensional, skill-centric representation that encodes which postural transitions are currently feasible. The high-level policy learns to select among skills within this space, allowing long-horizon locomotion to emerge through the coordination of developmentally grounded postural transitions.

To evaluate our hypothesis, we first validate the hierarchical framework in the LunarLander and BipedalWalker environments to isolate core components of skill decomposition and high-level coordination, independent of the developmental motivation. We then apply the framework to the Humanoid environment, where the developmental structure directly informs the task decomposition and exploration challenges are substantially greater. We find that in LunarLander and BipedalWalker, our framework achieves near-perfect task success, whereas monolithic RL and HRL baselines succeed only inconsistently within the same training budget. In the Humanoid setting, where the developmental decomposition is directly instantiated, our approach achieves upright posture in over 90% of evaluation episodes and executes the full developmental sequence in over 85%. In comparison, both monolithic and HRL baselines plateau at intermediate torso heights, with no method progressing to a fully upright standing posture. Our findings suggest that developmentally grounded structural priors substantially reduce the exploration burden of complex locomotion learning, enabling reliable postural progression where reward engineering, alternative exploration mechanisms and emergent hierarchies fall short.

Chapter 2

Background and Related Work

In this section, we begin by defining reinforcement learning (RL) as a formal framework for sequential decision-making and establish that getting up is a fundamental challenge in RL-based humanoid control. We then introduce hierarchical reinforcement learning (HRL) as a framework for decomposing complex tasks across temporal scales and review existing HRL approaches, emphasizing that most focus on discovering hierarchical structure and skill decompositions from scratch. Next, we examine what properties make individual skills effective once they are in hand, drawing on work that examines the roles of exploration, alignment and state abstraction. Finally, we turn to developmental psychology and describe the structured sequence through which humans acquire motor skills, which we argue provides a principled skill decomposition for HRL.

2.1 Reinforcement Learning

Reinforcement learning (RL) is a framework in which an agent learns to maximize cumulative reward through trial-and-error interactions with an environment. We for-

malize this as a continuous-state, continuous-action Markov Decision Process [14, 15]:

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho_0, \gamma \rangle,$$

where $\mathcal{S} \subseteq \mathbb{R}^n$ is the continuous state space, $\mathcal{A} \subseteq \mathbb{R}^m$ is the continuous action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the deterministic transition function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\rho_0 \in \Delta(\mathcal{S})$ is the initial state distribution and $\gamma \in [0, 1)$ is the discount factor. At timestep t, the agent observes state $s_t \in \mathcal{S}$, selects action $a_t \sim \pi(\cdot \mid s_t)$ according to policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, and receives reward $r_t = \mathcal{R}(s_t, a_t) \in \mathbb{R}$ while transitioning to state $s_{t+1} \in \mathcal{S}$. The objective is to find a policy that maximizes the expected discounted return:

$$J(\pi) = \mathbb{E}_{s_0 \sim \rho_0, \pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right].$$

To reason about the quality of states and actions, we define two key quantities. The state-value function $V^\pi(s)$ and action-value function $Q^\pi(s, a)$ measure expected return from a given state or state-action pair under π, defined as:

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s \right], \quad Q^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a \right].$$

These two functions are related by the equation:

$$V^\pi(s) = \mathbb{E}_{a \sim \pi(\cdot|s)}[Q^\pi(s, a)],$$

as the value of a state can be expressed as the expected value of the actions taken from it. The goal of RL is to find the optimal policy:

$$\pi^\star = \arg\max_\pi J(\pi) = \arg\max_\pi \mathbb{E}_{s_0 \sim \rho_0}[V^\pi(s_0)],$$

where the second equality follows from the fact that $J(\pi)$ is precisely the expected value of the initial state under π. The associated optimal value functions are

$$V^\star(s) = \max_\pi V^\pi(s), \quad Q^\star(s, a) = \max_\pi Q^\pi(s, a).$$

In discrete action spaces with bounded rewards, it can be shown that there exists a deterministic optimal policy [15] that can be recovered greedily as:

$$\pi^\star(s) = \arg\max_{a \in \mathcal{A}} Q^\star(s, a).$$

In continuous action spaces, however, computing the maximizing action at each step is intractable, motivating direct policy optimization instead. We overview two families of RL algorithms used in this work: value-based methods, which learn $Q^\star$, and policy-based methods, which optimize $J(\pi)$ directly.

2.1.1 Value-Based Methods

Value-based methods learn an approximation of $Q^\star$ and derive a policy by acting greedily with respect to it. Q-learning [16] is the foundational value-based algorithm, updating action-value estimates via:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right].$$

To extend Q-learning to high-dimensional state spaces, Deep Q-Networks (DQN) [17] parameterize Q with a neural network $Q_\phi(s, a)$, which is trained by minimizing the temporal difference loss:

$$L_{\text{DQN}}(\phi) = \mathbb{E}_t \left[\left(r_t + \gamma \max_{a'} Q_{\phi^-}(s_{t+1}, a') - Q_\phi(s_t, a_t) \right)^2 \right],$$

where ϕ^- denotes the parameters of a target network that is periodically updated to stabilize training. However, both Q-learning and DQN suffer from overestimation bias, as the max operator and network respectively are used to both select and evaluate actions. Any upward noise in Q-values is therefore preferentially selected and incorporated into the update, which can systematically inflate estimates. Double Q-learning [18] addresses this by decoupling action selection and evaluation, and Double DQN (DDQN) [19] instantiates this idea within the DQN framework: the online network Q_ϕ selects an action while the target network Q_{ϕ^-} evaluates it, so errors in one

do not directly amplify the other. The regression target for the loss is then:

$$y_t = r_t + \gamma Q_{\phi^-}\left(s_{t+1}, \arg\max_{a'} Q_\phi(s_{t+1}, a')\right).$$

The DDQN loss is:

$$L_{\text{DDQN}}(\phi) = \mathbb{E}_t\left[(Q_\phi(s_t, a_t) - y_t)^2\right].$$

DDQN is particularly well-suited to discrete action spaces, where the optimal action can be identified by taking the arg max over Q-values.

2.1.2 Policy-Based Methods

While value-based methods derive a policy through Q-values, policy-based methods treat the policy itself as the primary object of optimization. However, computing $\nabla J(\pi)$ is intractable, as it requires differentiating through unknown environment dynamics. Instead, we parameterize π as a neural network π_θ that takes a state as input and outputs a distribution over actions. We then optimize $J(\theta) \triangleq J(\pi_\theta)$ using the policy gradient theorem [20], which expresses the gradient as an expectation that can be estimated from sampled trajectories:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a_t \mid s_t) A^{\pi_\theta}(s_t, a_t)\right],$$

where $A^{\pi_\theta}(s, a) = Q^{\pi_\theta}(s, a) - V^{\pi_\theta}(s)$ is the advantage function measuring how much better action a is relative to the average action in state s under π_θ [21].

A key drawback of vanilla policy gradient methods is that gradient updates can be large and destabilizing, as a single poor update can collapse the policy. A secondary concern is data efficiency, as each batch of experience is discarded after a single gradient step. To address both, Proximal Policy Optimization (PPO) [22] uses a clipped surrogate objective that constrains how much the policy can change per update, enabling multiple epochs of updates on each collected batch without desta-

bilizing training. This objective is defined as:

$$L_{\text{PPO}}(\theta) = \mathbb{E}_t \left[\min(p_t(\theta)\hat{A}_t,\ \text{clip}(p_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right]$$

where

$$p_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$

is the probability ratio between updated and old policies. $\hat{A}_t$ is an estimate of the advantage function, for which we use Generalized Advantage Estimation (GAE) [23]:

$$\hat{A}_t = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \delta_{t+l}, \quad \delta_t = r_t + \gamma V_\phi^{\pi_\theta}(s_{t+1}) - V_\phi^{\pi_\theta}(s_t),$$

where $\lambda \in [0, 1]$ is the GAE decay parameter and T is the rollout horizon. The value function $V_\phi^{\pi_\theta}$ is trained by minimizing

$$L_V(\phi) = \mathbb{E}_t \left[(V_\phi^{\pi_\theta}(s_t) - \hat{R}_t)^2 \right],$$

where $\hat{R}_t = \hat{A}_t + V_\phi^{\pi_\theta}(s_t)$ is the GAE return estimate. L_{PPO} and L_V are optimized jointly in an actor-critic framework to train both the actor π_θ and critic $V_\phi^{\pi_\theta}$.

2.2 The Challenge of Learning to Get Up and Walk

Having formalized the RL framework, we now turn to the focus of this work: the challenge of learning to get up and walk. This task requires exploring a high-dimensional state space where most action sequences lead to failure, and meaningful progress toward standing often requires extended sequences of coordinated movement before any reward signal is received. Unlike the periodic locomotion phase that follows, the getting up phase in particular lacks structured contact patterns, making it difficult to leverage domain-specific knowledge about effective gait patterns and often resulting in unnatural movement strategies [5].

Several recent works have explicitly addressed this problem. He et al. [4] develop HUMANUP, a two-stage pipeline for real-world humanoid robots that first discovers get-up trajectories under minimal deployment constraints before refining these into

smooth, torque-safe policies suitable for real-world deployment. While effective at addressing real-world challenges such as terrain variation and motor safety, the resulting motions adopt unconventional strategies such as backbends and highly dynamic movements. Tao et al. [5] explicitly target more natural get-up motions, proposing a three-stage framework that discovers motions with a strong agent, then progressively constrains torque limits and motion speed. While their approach improves naturalness by limiting physical capabilities, both methods ultimately discover movement strategies through exploration rather than imposing structure based on how humans actually move. Neither specifies which intermediate postural configurations the agent should pass through, leaving this decomposition to emerge through trial and error.

2.3 Hierarchical Reinforcement Learning

One particularly challenging requirement of the task of learning to walk, and indeed of many real-world tasks more broadly, is the need to reason and act across multiple temporal scales. For example, an agent may discover how to briefly achieve an upright position, but maintaining it requires making decisions that play out over hundreds of steps. In such settings, relying solely on primitive, one-step actions can often make RL slow and brittle, especially in environments with large state spaces or sparse rewards. Hierarchical reinforcement learning (HRL) offers a powerful framework to address this challenge by introducing temporal abstraction, or the ability for agents to operate using multi-step chunks of behavior rather than individual actions [8]. While several formalisms have been proposed to implement this idea [24, 25, 26], we focus here on the options framework [9].

Formally, we introduce a finite set of temporally extended options $\mathcal{O} = \{o_1, \ldots, o_K\}$. Each option $o \in \mathcal{O}$ is defined by three components: an intra-option policy $\pi_o : \mathcal{S} \to \Delta(\mathcal{A})$ that selects actions while o is active, a termination function $\beta_o : \mathcal{S} \to [0, 1]$

giving the probability option o terminates upon reaching state s, and an initiation set $I_o \subseteq \mathcal{S}$ specifying the states from which o may be invoked.

When option o is selected in state s_t, its intra-option policy π_o is executed until termination, producing a trajectory segment of length $\tau \geq 1$ and yielding a transition to state $s_{t+\tau}$. Because τ can vary across options and states, decisions are made at irregular, option-determined time intervals rather than at every discrete step. The resulting process is a semi-Markov decision process (SMDP) over options defined as:

$$\mathcal{M}^{\text{SMDP}} = (\mathcal{S}, \mathcal{O}, P^O, R^O, \gamma),$$

where

$$P^O(s'|s, o) = Pr(s_{t+\tau} = s'|s_t = s, o)$$

is the probability of transitioning to state s' after executing option o from state s until termination and

$$R^O(s, o) = \mathbb{E}\left[\sum_{k=0}^{\tau-1} \gamma^k r(s_{t+k}, a_{t+k})|s_t = s, o\right]$$

is the expected discounted reward accumulated over the option's execution. The high-level policy thus operates over this induced SMDP, selecting among options rather than primitive actions.

We restrict optimization to a hierarchical policy class defined by a high-level policy $\mu_\phi(o|s)$ over options and a set of intra-option policies $\{\pi_{\theta_o}\}_{o\in\mathcal{O}}$:

$$\Pi_{\text{HRL}} = \{\pi : \pi \text{ is induced by } (\mu_\phi, \{\pi_{\theta_o}\}_{o\in\mathcal{O}})\}.$$

The hierarchical objective is to find a high-level policy and set of intra-option policies that jointly maximize expected discounted return:

$$\max_{\phi, \{\theta_o\}_{o\in\mathcal{O}}} J(\mu_\phi, \{\pi_{\theta_o}\}) = \mathbb{E}_{\mu_\phi, \{\pi_{\theta_o}\}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right].$$

In our framework, the intra-option policies $\{\pi_{\theta_o}\}$ are pretrained and held fixed, and we optimize only the high-level policy parameters ϕ.

2.4 Existing Hierarchical RL Approaches

Having established the options framework as our formal basis for hierarchical control, we now survey existing HRL methods and their relationship to our approach. Across this work, the key idea is the use of temporal abstraction to decompose complex tasks into subproblems in order to reduce the burdens of exploration and credit assignment. Dayan and Hinton [24] formalize this intuition by organizing control into a multi-level hierarchy of managers and submanagers. In this framework, higher-level managers set subgoals for the submanagers at the level below, who are then rewarded solely for satisfying those commands without needing to understand the global objective. Each level thus operates at its own temporal resolution, enabling high-level components to reason over longer horizons while delegating fine-grained execution to lower levels. Two principles enforce this modularity: reward hiding, where submanagers are rewarded only for accomplishing their assigned subgoals independently of the global objective, and information hiding, where each level observes only the state information relevant to its own decisions. Together, these constraints enable more scalable credit assignment, since each level only optimizes over its own subgoal, and more efficient exploration, since subgoals narrow the space each level must search.

While the options framework provides a powerful formalism of temporal abstraction, early implementations assumed a fixed set of pre-defined options. Bacon et al. [12] derive policy gradient theorems for both intra-option policies and termination conditions, allowing options to emerge autonomously from only the high-level task reward. In this framework, an option is nudged toward terminating whenever continuing it yields lower expected value than switching. In grid-world navigation and Atari tasks, this produces an interpretable structure, with distinct options specializing into semantically coherent behaviors. However, in settings where the task reward is sparse or the state-action space is large, this reliance on the task reward can create a coupling problem: the agent cannot learn when to switch options without already having

coherent options to execute, and it cannot learn coherent options without meaningful switching behavior. In high-dimensional continuous control, this circularity is especially acute, since early-training reward signal is nearly absent.

Vezhnevets et al. [11] subsequently revisited the feudal perspective, extending the original framework to high-dimensional, partially observable environments by incorporating deep neural networks for both goal-setting and control. In this framework, a high-level manager emits latent vectors in a learned embedding space, and a low-level worker is trained to align its actions with these goals. In Atari games and discrete, image-based tasks with long reward delays, this architecture is thus able to leverage temporally abstracted goals to help propagate sparse rewards across extended horizons. However, because both the goal representation and the skill-conditioned policies must be learned jointly from high-dimensional observations, this approach places substantial demands on representation learning and exploration. The manager operates in a latent space it is simultaneously constructing, meaning that the worker has no stable target to learn from during early training, and discovering a useful goal space can become a significant source of sample complexity.

Nachum et al. [10] (HIRO) addresses this key limitation of the feudal approach and extends HRL to continuous control domains. Rather than operating in a learned latent space, HIRO's high-level controller emits goals defined as desired relative changes in raw observation space, immediately giving the lower level a dense intrinsic reward signal. To enable off-policy training at both levels, HIRO introduces a goal labeling scheme that corrects for the non-stationarity introduced as the lower-level policy evolves, in which past high-level transitions are relabeled with the goal that would have made the observed low-level actions most likely under the current policy. On MuJoCo benchmarks requiring both coordinated locomotion and object interaction (including Ant Maze, Ant Push and Ant Fall), this proves substantially more effective than prior HRL methods. We note that these tasks benefit from reachable

goals that are dense and well-distributed in joint space, as the ant morphology is quadrupedal and inherently stable, making random goal-directed exploration informative. For humanoid locomotion, however, this assumption does not necessarily hold. The reachable state space is vast and much of it dynamically unstable, so undirected high-level exploration over raw observations provides little useful signal about which state transitions constitute coherent motor primitives.

A common thread across these methods is that hierarchical structure is something to be discovered, as the agent must identify useful temporal abstractions through interaction alone. This works when task reward provides sufficient signal to shape the decomposition, but this premise may break down in high-dimensional continuous control, where early training feedback is sparse and the space of possible behaviors is large. Rather than leaving the hierarchical decomposition to be discovered, our approach grounds it in the developmental structure of human motor learning, constraining the space of skill sequences that the high-level agent needs to consider.

2.5 Designing Good Options

Beyond the question of how hierarchical structure is discovered, a separate line of work examines what properties make options effective once they are in hand. Sutton et al. [9] illustrate this in a four-room Gridworld using "hallway" options designed to reliably reach doorways between rooms. When the task goal aligns with these subgoals, option-based learning maintains a clear advantage over primitive-action learning. They attribute this performance gain to more directed exploration, as rather than diffusing probability mass across the full action space, options steer the agent toward subgoal states, concentrating experience where it is most useful. When goals and subgoals are misaligned, however, hallway options perform worse because the optimal solution requires action sequences that the options cannot express. This sug-

gests that the value of an option set depends not only on how well individual options are designed, but on whether the set as a whole covers the relevant task structure.

Jong et al. [27] further analyze the utility of temporal abstraction through the lens of exploration, arguing that how options are integrated into the action space determines their impact on performance. When options augment the primitive action set, random exploration becomes biased toward option termination states. If those states are far from the reward, a single option selection can therefore undo several steps of primitive exploration, making it increasingly difficult for the agent to reach the task reward. When options instead replace primitive actions and are well-designed, they constrain exploration in a way that substantially accelerates learning by preventing the agent from exploring irrelevant parts of the state space entirely. In their four-room experiments, the agent that uses options in the three non-goal rooms and primitive actions only inside the goal-room learns fastest precisely because the options eliminate three rooms from the exploration problem. This suggests that a central benefit of temporal abstraction lies in the structure it imposes on where the agent explores.

Beyond exploration, the efficiency of HRL also depends on the interaction between temporal abstraction and state abstraction. Konidaris [28] argues that solving complex tasks requires the joint construction of action abstractions, or options, and task-specific state abstractions. In particular, temporally extended skills naturally induce compact state representations tailored for higher-level reasoning, and without such representations, the higher-level policy faces an unnecessarily large and poorly structured state space. Sutton et al. [9] gesture at a related point, drawing a connection between temporally extended action and the potential for temporally extended "perception." Just as options allow an agent to act over extended time horizons, they may similarly allow it to perceive and represent the world at a coarser temporal grain.

A complementary perspective is offered by Dietterich [25], which assumes that a useful task decomposition is provided and studies how that structure can accelerate

learning. Their algorithm, MAXQ, represents hierarchy explicitly as a task graph of subtasks and derives a corresponding decomposition of the value function across levels, enabling state abstractions within each subtask that substantially improve learning efficiency. The framework requires the designer to supply the task graph, meaning it does not address where an appropriate decomposition should come from. What MAXQ does demonstrate, however, is that a principled decomposition is worth having, as the gains from well-structured hierarchies are substantial enough to motivate the harder question of how to obtain one.

Taken together, this work support two claims that guide our approach. First, temporal abstraction can meaningfully accelerate learning when options are well-aligned with the high-level task by imposing structure on where the agent explores. Second, the effectiveness of HRL depends not only on the quality of individual options but on the joint coordination of action and state abstraction, as skills must induce representations that make higher-level reasoning tractable. To address the former, we turn to developmental psychology to define a principled task decomposition.

2.6 Human Motor Development

The preceding sections establish that temporal abstraction can meaningfully accelerate learning, but only when the skill decomposition is well-matched to task structure. This raises a prior question: where should a principled decomposition come from? Turning to cognitive science, work on human learning suggests that the hierarchies that humans employ are not arbitrary. Rather, human learners discover task decompositions that carve structure at its natural boundaries, favoring hierarchical representations that minimize the complexity of encoding adaptive behavior [29]. For the problem of getting up and walking, we turn to developmental psychology.

The developmental psychology literature offers a principled decomposition of lo-

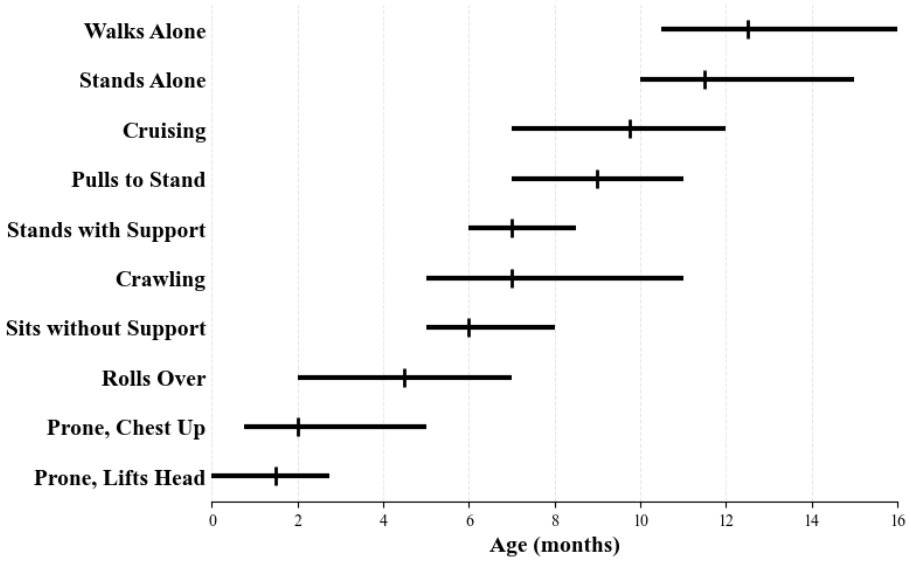


Figure 2.1: Human Motor Development Timeline, adapted from [6].

comotion into a sequence of postural control problems, each with distinct biomechanical demands and its own learning curve. Figure 2.1 illustrates a canonical timeline of early motor development, highlighting the sequence of postural transitions that precede independent walking [1, 6]. Infants begin by developing head and trunk control in prone positions, first lifting their heads, pushing up with their arms and rolling over. As they gain strength, infants sit without support, crawl, stand with support and pull to stand. With greater stability, they progress to cruising, standing alone and ultimately walking alone. While this sequence is widely recognized as a standard, motor learning remains flexible: infants may skip stages, reach milestones at different times or adopt alternative strategies depending on environmental and cultural factors [6]. Developmental motor research therefore does not prescribe a single invariant sequence but rather suggests the set of postural control problems that are commonly solved on the path to upright locomotion.

A key insight is that postures are not arbitrary waypoints. Stable posture provides the foundation for exploration, manipulation and locomotion, and increasingly upright postures are acquired by progressively reducing the base of support while

raising the center of mass [1]. Each postural *transition* is therefore especially meaningful, changing the biomechanical demands of control, creating new bases of support and opening novel opportunities for perception and action. Indeed, developmental evidence suggests that learning is highly posture-specific, as experience in one posture does not automatically transfer to another. For example, infants who are experienced crawlers or cruisers must still learn the unique balance demands of walking [1, 30]

Adolph [7] characterizes this process as "learning to learn." Within each posture, infants acquire the ability to use perceptual information relevant to that posture-specific control problem, rather than learning fixed facts or solutions that generalize across postures. Each postural configuration defines a distinct problem space with its own relevant parameters, exploratory behaviors and learning curve [7]. The developmental evidence thus motivates treating each postural transition as a genuinely distinct subproblem whose boundaries are defined by key discontinuities in the underlying control problem. This is precisely the structure the options framework is designed to exploit: each posture-specific control problem maps naturally onto an intra-option policy with a well-defined initiation context, and discontinuities between postures provide principled boundaries at which a high-level controller can operate.

Rather than modeling the full developmental timeline shown in Figure 2.1, we focus on the subset of postural transitions directly relevant to the problem of getting up from the ground and achieving unsupported locomotion. These transitions form a structured path from contact-rich, stable configurations to upright bipedal locomotion, with each stage incrementally reducing external support, narrowing the base of support and increasing the demands on balance, coordination and weight transfer.

Key Postural Transitions

Early locomotion emerges in prone configurations that minimize balance demands and maximize stability through continuous ground contact. Infants adopt a variety

of prone strategies, including pivoting and belly crawling, which rely on distributed support across the body and enable movement without requiring extensive balance [1, 7]. Transitioning to hands-and-knees crawling introduces a qualitatively different control problem as the torso is lifted off the ground, requiring coordinated limb activity to maintain balance [7]. The prone-to-crawl transition captures the shift from fully supported, low-dimensional movement to coordinated, self-supported locomotion.

The transition from crawling to kneeling represents a further reorganization of support that begins to approximate upright posture. Moving from crawling to kneeling removes two points of ground contact and requires greater control of the trunk and hips to stabilize the body over a smaller support base [1, 31]. The crawl-to-kneel transition thus marks a shift from using the limbs for distributed support to bearing body weight more directly in preparation for upright stances.

The importance of the kneel-to-lunge and lunge-to-stand positions is strongly supported by studies of pulling-to-stand. Infants do not transition directly from the symmetric support of kneeling to upright standing. Instead, they quickly learn to adopt an asymmetric half-kneel strategy, in which one leg kneels while the other plants at a 90-degree angle and pushes into extension [31, 32]. This posture is qualitatively distinct from symmetric kneeling, requiring dissociation of the lower limbs and an asymmetric lateral weight shift. Completing the rise then requires a further transition from the half-kneel posture to full upright standing, in which body weight transfers entirely onto the feet and balance must be maintained with minimal ground contact.

Finally, walking introduces a new control problem in which balance must be maintained dynamically over alternating single-limb support phases [1, 7]. We note that cruising—sideways locomotion while holding furniture—is an important transitional behavior. However, empirical evidence shows that it represents a functionally distinct skill that does not transfer to upright walking, as control is primarily organized around the arms rather than the legs [30]. For this reason, cruising falls outside the

scope of our work, which focuses on unsupported transitions to bipedal locomotion.

Taken together, these transitions—prone-to-crawl, crawl-to-kneel, kneel-to-lunge, lunge-to-stand and stand-to-walk—form a structured progression from contact-rich, stable configurations with a low center of mass to dynamically stable bipedal locomotion. Each stage introduces a qualitatively new control requirement—whether distributed support across limbs, trunk control over a reduced base of support, asymmetric weight transfer, full bipedal support or dynamic balance—such that removing any stage would skip a necessary reorganization of the control problem.

2.7 Summary

Learning to get up and walk is a challenging task for humanoid control, requiring extended sequences of coordinated movement in a high-dimensional state space where most action sequences lead to failure. HRL offers a way to decompose this challenge across temporal scales, reducing the credit assignment burden by organizing behavior into reusable, temporally extended options. The HRL methods reviewed above largely share a common approach in discovering hierarchical structure from scratch. While this flexibility enables generalization across tasks, it places substantial demands on exploration and provides no guarantee that the resulting decompositions will be aligned with task structure or data-efficient. Furthermore, the benefits of temporal abstraction depend critically on whether options are well-matched to task structure and coordinated with an appropriate state abstraction.

The developmental psychology literature offers an alternative starting point to discovering an emergent hierarchy by documenting the structured sequence of postural milestones through which humans reliably acquire motor skills. We adopt this sequence as a principled decomposition of the full locomotion learning problem into a set of stable, behaviorally meaningful subproblems.

Chapter 3

Approach

The key novelty of our work lies in using human motor development as a structural prior for HRL. Most existing HRL methods are designed with generality in mind, as by discovering skill decompositions through interaction, they can in principle adapt to a wide range of tasks. This generality comes at a cost: without prior knowledge of which decompositions are useful, the agent must explore a vast space of possible hierarchies, and there is no guarantee that the resulting structure will be aligned with task-relevant boundaries or data-efficient.

For the specific problem of learning to stand and walk, we argue that this generality is unnecessary. The developmental sequence through which humans acquire upright locomotion is well-documented, stable across individuals and grounded in biomechanical structure [1, 6, 30]. Rather than discovering a skill set through exploration, we impose this developmental decomposition directly, constraining the space that the high-level policy must search over and providing a coherent starting point for learning from training onset. While this makes our approach narrower in scope than general-purpose HRL methods, we argue that this is a feature rather than a limitation, as the developmental prior is task-aligned and directly addresses the exploration challenges that make humanoid locomotion difficult. We also note that the

broader strategy of instantiating a known curriculum as a fixed option set within HRL is not specific to locomotion. Thus, our approach is extensible to any domain where a principled task decomposition can be identified from prior knowledge.

In this section, we describe how we instantiate this developmental structure within an HRL framework. We begin by formalizing the developmental sequence as a set of options. We then introduce the Value Function Spaces (VFS) framework of Shah et al. [13], which constructs a skill-centric state abstraction for high-level decision-making, and describe how we instantiate it using our developmental options set. Finally, we discuss how the imposed developmental hierarchy constrains the agent's exploration space, channeling learning along the same stable pathway that humans follow when acquiring upright locomotion.

3.1 Developmental Transitions as Options

As established in Section 2.6, human motor development proceeds through a well-defined sequence of postural milestones. We focus on the subset of postural transitions directly relevant to getting up and achieving unsupported locomotion: prone-to-crawl, crawl-to-kneel, kneel-to-lunge, lunge-to-stand and stand-to-walk. Within this sequence, each posture corresponds to a biomechanically stable configuration and introduces a qualitatively distinct control requirement. Critically, humans do not solve these transitions jointly, as developmental evidence suggests that each is acquired as a separate learning problem, with its own exploratory behaviors, parameters and learning curve [7]. This is precisely the structure that options are designed to exploit: temporally extended, task-aligned subproblems whose boundaries correspond to meaningful discontinuities in the control problem.

We therefore formalize this developmental progression using the options framework introduced in Section 2.3. Each postural transition is treated as a distinct option

$o \in \mathcal{O}$, where:

$$\mathcal{O} = \{o_{\text{prone-to-crawl}},\ o_{\text{crawl-to-kneel}},\ o_{\text{kneel-to-lunge}}, o_{\text{lunge-to-stand}},\ o_{\text{stand-to-walk}}\}.$$

The intra-option policy π_o defines the low-level motor control strategy for executing each transition. This formulation differs fundamentally from previous HRL methods, which must discover both the skill set and the skill boundaries through interaction—a circular problem in which meaningful options cannot emerge without reward signal and meaningful reward signal cannot propagate without coherent options. By specifying the option set directly from human motor development, we bypass this discovery problem entirely, enabling the high-level policy to inherit a skill decomposition that is aligned with task structure from the start of training. We note that each phase of the developmental curriculum induces a distinct termination distribution over states, corresponding to the completion of a particular postural configuration. This structure is itself informative, and future work could exploit these termination distributions as a starting point for more data-driven approaches, such as by learning termination functions directly from the state distributions at skill boundaries [33].

3.2 Value Function Spaces for Skill Composition

While the developmental hierarchy defines which skills exist, the agent still faces the problem of deciding which skill to execute at any given moment. A naive approach might operate directly over the full humanoid state space, which includes joint angles, joint velocities, center of mass velocities and contact forces. As Konidaris [28] argues, however, effective hierarchical control requires both action abstraction and a corresponding state abstraction that exposes only the information critical to the high-level decision. Without such a representation, the high-level policy inherits the full complexity of the low-level state space, including potentially task-irrelevant details, thus undermining the efficiency gains that temporal abstraction is meant to provide.

The Value Function Spaces (VFS) framework addresses this by constructing a compact, skill-centric state representation from the value functions of pretrained skills [13]. Given a set of options $\mathcal{O}$ with pretrained policies $\{\pi_k\}$ and associated value functions $\{V_k\}$, VFS constructs an embedding by stacking these value functions:

$$Z(s) = [V_1(s), V_2(s) \ldots, V_{|\mathcal{O}|}(s)].$$

The central insight is that a skill's value function $V_k(s)$ captures not just the expected return under that skill but also an implicit measure of whether executing skill k from state s is likely to succeed. A high value indicates that the current state affords the transition that skill k is designed to perform, while a low value suggests the skill is not currently executable or would lead to failure.

Stacking these value functions thus compresses the high-dimensional state s into a low-dimensional representation that captures which skills are currently viable, providing a natural basis for high-level decision-making. Crucially, $Z(s)$ exhibits useful invariance properties, as states that differ in task-irrelevant details but afford the same set of transitions will have similar embeddings, while states that support different transitions will be well-separated in embedding space.

3.3 Instantiating the Developmental Hierarchy

We instantiate the VFS framework using the developmental option set $\mathcal{O}$ defined in Section 3.1. After pretraining each intra-option policy π_o with its skill-specific reward and initial state distribution, we obtain a corresponding value function $V_o(s)$ and construct the skill-centric embedding:

$$Z(s) = [V_{\text{prone-to-crawl}}, V_{\text{crawl-to-kneel}}, V_{\text{kneel-to-lunge}}, V_{\text{lunge-to-stand}}, V_{\text{stand-to-walk}}].$$

The high level policy μ operates over this embedding rather than the raw state, observing $Z(s)$ at each decision point and selecting the next option to execute. Because

$Z(s)$ is low-dimensional, the high-level policy can learn to sequence skills efficiently without reasoning about the full complexity of humanoid kinematics.

Training is structured in two phases. First, each intra-option policy is independently pretrained with skill-specific rewards and initial state distributions derived from the corresponding postural transition. These policies are then frozen, and the high-level controller is trained to compose them using either DDQN, which treats option selection as a discrete Q-value problem over $\mathcal{O}$, or PPO, which we evaluate as an alternative high-level algorithm due to its stability in continuous control settings.

This approach bears structural resemblance to SayCan [34], which uses value functions alongside large language model scores to select skills for robotic manipulation. Where SayCan's prior over skill selection comes from a large language model, however, ours is grounded in developmental psychology, with the structured progression of human motor milestones constraining the space of valid skill compositions.

3.4 Developmental Constraints and Exploration

Our imposed developmental hierarchy has direct consequences for how the agent explores the state space. In a generic RL formulation, the agent must search through the entirety of the high-dimensional observation space without any prior over which regions are worth visiting. For humanoid locomotion, this is especially acute, as most states correspond to falls or dynamically unstable configurations, and the reward signal needed to begin learning is unlikely to be encountered through undirected exploration alone. Existing HRL methods partially address this by decomposing behavior across temporal scales, but they introduce their own exploration challenges, as HRL agents must simultaneously discover what skills to learn and how to sequence them. Without a prior on either, the high-level policy wastes experience on skill combinations that are meaningless or lead to immediate failure, while the low-level policies

waste experience searching for skills that may not constitute useful motor behavior.

Our developmental hierarchy addresses these exploration challenges in two complementary ways. First, because the option set is specified directly by developmental structure, low-level policies are trained with skill-specific rewards and initial state distributions, ensuring that each skill explores the subspace of states relevant to its transition. This dramatically reduces the effective dimensionality of the exploration problem at the skill level. The resulting skills then constrain the exploration space for the high-level policy, preventing the agent from wasting experience in regions that do not contribute to task progress. Second, the developmental sequence naturally constrains valid skill orderings, as each skill is trained to operate from a specific postural configuration, and invoking a skill outside its intended context is likely to fail immediately, particularly given the complex dynamics of the Humanoid environment. The VFS embedding reinforces this by encoding the feasibility of each skill at the current state, providing the high-level policy with a stronger signal during skill selection.

Taken together, these constraints eliminate vast regions of the policy space that would lead to immediate failure, instead directing exploration along the same stable, incremental pathway that humans follow when learning to walk. Our hypothesis is that this developmental structure yields improvements in sample complexity and robustness relative to both flat RL baselines and hierarchical methods that discover their own skill decomposition from scratch.

Chapter 4

Experimental Overview

In this chapter, we describe the experimental setup, baseline comparisons and evaluation methodology used to assess our framework.

4.1 Experiments

We evaluate our approach across three environments of increasing complexity [35, 36]. We begin with a modified version of `LunarLander`, which has simple dynamics and a discrete action space, making it a tractable testbed for validating our approach. We next evaluate in `BipedalWalker`, a continuous-control environment with a 24-dimensional state and 4-dimensional continuous action space, where coordinating motor primitives becomes more difficult. Finally, we apply our framework to the full `Humanoid` environment, where the high-dimensional, continuous state-action space makes exploration especially challenging and the developmental hierarchy directly informs the task decomposition. Together, the three environments allow us to validate the framework incrementally, first establishing skill coordination before approaching the full complexity of humanoid motor control.

4.2 Baselines

We compare against two categories of baselines: monolithic (flat) PPO variants with different reward functions and exploration mechanisms, and an HRL method that learns an emergent skill decomposition end-to-end. We describe these baselines below and summarize the experimental question they isolate in Table 4.1. Specific baseline details for each environment are provided in the corresponding experiment chapter.

4.2.1 Flat PPO Baselines

All flat PPO variants learn a single monolithic policy over the full action space, differing only in how reward and exploration are structured.

Sparse Reward receives a reward signal only upon task completion, providing no intermediate feedback. This is the most conservative baseline, establishing whether the task is solvable without any structural guidance.

Intermediate Goals receives signal at each skill milestone, providing the same structural knowledge encoded in our task decomposition but without enforcing separate policies at each level. This tests whether knowing what milestones matter is sufficient, or whether the constraint of learning dedicated skills is independently necessary.

Distance Reward receives a continuous reward proportional to progress toward the goal state, with no intermediate milestone structure. This tests whether dense, continuous feedback alone substitutes for our decomposition, or whether explicitly decomposing the task into ordered sub-behaviors is useful.

Random Network Distillation (RND) augments the sparse reward baseline with an intrinsic exploration bonus that incentivizes visiting novel states [37], instantiating a broader class of approaches that address sparse reward either through reward shaping [38] or intrinsic motivation [39]. Unlike our hierarchy, which channels exploration through developmental skills, RND's prior over which regions of the state space are

worth visiting favors only novelty, with no preference for which of those unfamiliar regions are task-relevant. This lets us test whether exploration quantity can substitute for our approach, or whether the directed structure of our task decomposition is the operative factor. We provide algorithmic details of RND in Appendix A.

4.2.2 HRL Baseline: HIRO-PPO

HIRO [10] serves as our primary HRL comparison, representing a strong prior approach for continuous control and directly testing whether our imposed developmental hierarchy outperforms an emergent one. In this algorithm, a high-level controller proposes subgoals as desired relative displacements in observation space every C steps, while the low-level policy receives an intrinsic reward proportional to how closely it achieves those goals. Both policies are trained off-policy via Deep Deterministic Policy Gradient [40] with a replay buffer, and a goal-relabeling correction accounts for the mismatch between the goal originally proposed and the goal the low-level policy actually pursued during stored transitions.

However, because our framework trains low-level skills with PPO, we implement an adapted version of HIRO that we call HIRO-PPO. This implementation maintains the original goal-proposal and intrinsic reward structure but conducts both high- and low-level policy updates on-policy, eliminating the need for the goal-relabeling correction. PPO has demonstrated strong performance in continuous control settings [22], and using a consistent optimization algorithm across both our framework and this baseline ensures that any performance differences reflects the decomposition rather than the optimizer. We provide implementation details of HIRO-PPO in Appendix A.

We note that other HRL baselines exist within the continuous-control domain, namely Option-Critic [12]. However, Option-Critic jointly optimizes both intra-option policies and termination conditions from task reward alone. This is a non-trivial problem [41], and it risks conflation with the question we wish to answer, as performance

differences could reflect the difficulty of learning termination conditions rather than the value of the developmental decomposition. HIRO-PPO provides a cleaner comparison by varying only whether the task decomposition is imposed or emergent.

Category	**Baseline**	**Experimental Question**
Flat PPO	Sparse Reward	Is the task solvable without any structural prior?
Flat PPO	Intermediate Goals	Are intermediate goal signals sufficient without our imposed hierarchy?
Flat PPO	Distance Reward	Is reward density sufficient without our imposed hierarchy?
Flat PPO	Random Network Distillation	Is intrinsic motivation sufficient without our imposed hierarchy?
HRL	HIRO-PPO	Does our imposed hierarchy outperform an emergent one?

Table 4.1: Overview of baselines and their corresponding experimental question.

4.3 Evaluation Methods

We evaluate on three axes: task success rate, sample complexity [42] and movement quality. Task success rate is measured as the fraction of evaluation episodes in which the agent achieves the target behavior within the episode horizon. Sample complexity is measured by the total number of steps required to reach a given performance threshold, including both low-level and high-level policy training. Finally, for the Humanoid environment only, we additionally consider movement quality metrics including center-of-mass stability, energy expenditure and qualitative assessments of learned postural progressions.

Chapter 5

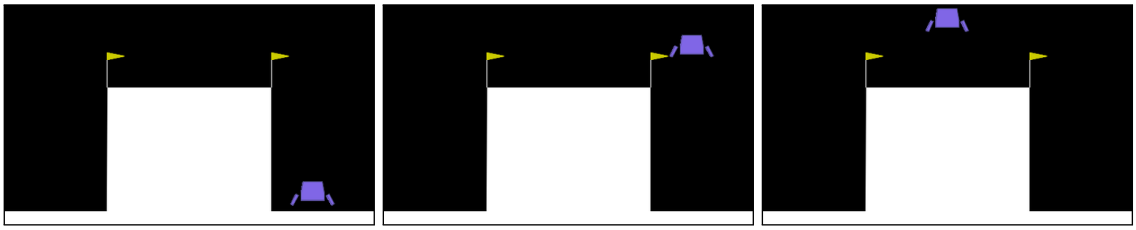
Lunar Lander Experiment

5.1 Methods

Before applying our framework to humanoid locomotion, we first validate its core components in the simpler LunarLander environment. This experiment serves as a controlled test of whether the high-level controller can learn to coordinate independently pretrained skills using only a sparse task reward and the VFS embedding.

5.1.1 Environment

We use a modified version of the LunarLander-v2 environment [35], which features a lander that operates in a 2D world with an 8-dimensional state space consisting of position, velocity, angle, angular velocity and leg contact and a 4-dimensional discrete action space. The standard task—descending from a fixed spawn position and landing between two flags—can be solved by a flat policy with a dense reward. To make the task more challenging and necessitate multi-stage behavior, we introduce a hard variant shown in Figure 5.1, in which the landing pad is placed on an elevated platform and the agent spawns on the ground to one side.



(a) Takeoff (b) Center (c) Land

Figure 5.1: Modified LunarLander environment and initial state distribution for each skill. The agent must take off, center and land between the flags.

5.1.2 Skill Design and Training

We define three low-level skills aligned with the modified task structure, as shown in Figure 5.1: takeoff, in which the agent lifts off the ground and maneuvers vertically upwards; center, in which the agent moves horizontally over the landing pad; and land, in which the agent descends carefully onto the landing zone. Each skill is trained using PPO with a skill-specific reward and initial state distribution, described below. Hyperparameters are provided in Appendix C.

Takeoff: The takeoff skill is initialized with the agent resting on the ground at a fixed horizontal position from the landing pad. Its reward function uses inverse-distance shaping toward a target position directly above the initial position, encouraging the agent to ascend vertically. Horizontal velocity is penalized to encourage stable ascent, and episodes terminate if the agent moves out of bounds or crashes.

Center: The center skill is initialized with the agent airborne above and to one side of the platform. Its reward function uses inverse-distance shaping toward a target position centered above the pad. Vertical velocity is penalized to encourage stable horizontal translation. Episodes terminate if the agent moves out of bounds or crashes.

Land: The landing skill is initialized with the agent positioned directly above the pad. Its reward function uses inverse-distance shaping toward the pad surface, with an additional bonus for simultaneous leg contact. A sparse success reward is given when both legs contact the ground within a tight horizontal tolerance and with sufficiently

low velocity. Episodes terminate if the agent lands, crashes or moves out of bounds.

5.1.3 High-level Policy Training

Given three trained skill policies, we train a high-level selector to choose which skill to execute at each decision point. The selector observes the VFS embedding $Z(s) = [V_{\text{takeoff}}(s), V_{\text{center}}(s), V_{\text{land}}(s)]$, where each component is the value estimated by the corresponding skill's frozen critic. It selects one of the three skills, which then executes for a fixed number of environment steps T_{option} before the selector is queried again. The selector receives a sparse reward of $+1000$ only upon successful landing and -100 upon termination (crashing or moving out of bounds), receiving no intermediate signal about skill ordering or transition quality. It uses the same initial state distribution as the takeoff skill, beginning on the ground.

We evaluate two selector algorithms—DDQN and PPO—and vary the option horizon $T_{\text{option}} \in \{50, 75, 100\}$, for a total of six conditions. In all cases, the high-level policy must discover the correct skill ordering—takeoff, center, land—using only the sparse landing reward and the structure implicit in $Z(s)$.

5.1.4 Baselines and Evaluation

We compare against the baselines described in Section 4.2, instantiated for the LunarLander environment. The sparse reward baseline receives a success signal only upon landing safely, as during selector training. The intermediate goal reward baseline receives incremental rewards at manually specified target positions, which correspond to the goal locations of each of our skills. The distance reward baseline receives continuous feedback proportional to proximity to the landing pad. The RND baseline augments the sparse reward with an intrinsic motivation bonus. Finally, we compare to HIRO-PPO under a dense distance reward. Hyperparameters are provided in Appendix C. We evaluate on task success rate, averaged across 5 training seeds and 5

evaluation episodes per seed, and report total environment training steps.

5.2 Results

5.2.1 Skills

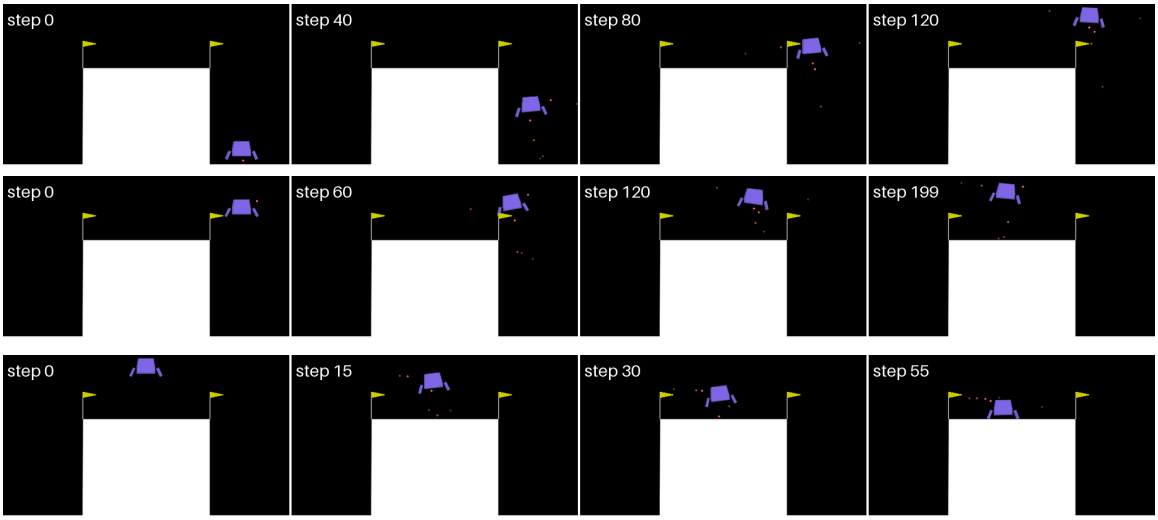


Figure 5.2: Filmstrips of learned skill policies on a representative episode in LunarLander. **Top:** Takeoff. **Middle:** Center. **Bottom:** Land.

Figure 5.2 shows a filmstrip of each skill's learned behavior across a representative episode. The takeoff skill learns the intended upward motion, the center skill moves horizontally over the landing platform and the landing skill descends carefully to the landing pad. As shown in Table 5.1, all three skills achieve a 100% evaluation success rate, confirming that the skill decomposition renders each subproblem tractable in isolation. We note that takeoff and center require slightly longer training than land, as their target regions are spatially adjacent and the policies must develop sufficiently distinct behaviors to avoid ambiguity in downstream skill composition.

Table 5.1: LunarLander skills performance across 5 training seeds, each averaged over 5 episodes. Success rate is defined as reaching the target (x, y) location.

Skill	**Steps (Millions)**	**Evaluation Success Rate**
Takeoff	0.8M	1.0
Center	0.8M	1.0
Land	0.4M	1.0

Table 5.2: LunarLander selector performance across algorithms and option horizons T_{option}. Success rate, defined as landing safely, is averaged over 5 training seeds, each evaluated on 5 episodes.

Algorithm	$T_{\textbf{option}}$	**Steps (Millions)**	**Evaluation Success Rate**
DDQN	50	0.3M	1.0
DDQN	75	0.25M	1.0
DDQN	100	0.3M	1.0
PPO	50	0.8M∗	1.0
PPO	75	0.4M	1.0
PPO	100	0.3M	1.0

∗Required extended skill training (0.2M additional steps for takeoff and center) for PPO $T = 50$, which is included in this quantity.

5.2.2 Selector

Given the three trained skill policies, we train a high-level selector that chooses which skill to execute at each decision point. Table 5.2 summarizes selector performance across algorithms and option horizons T_{option}. Both DDQN and PPO achieve 100% success across all values of T_{option}, with DDQN converging in as few as 0.25M steps and PPO requiring up to 0.4M steps. The exception is PPO at $T_{\text{option}} = 50$, which required 0.2M additional training steps for the takeoff and center skills beyond the standard budget, reflected in the 0.8M total reported.

Figure 5.3 shows the resulting trajectories across all conditions. Both algorithms produce consistent behavior: the lander rises from the platform (takeoff, blue), traverses horizontally to align with the pad (center, green) and descends onto it (land, red). Notably, the selector recovers the correct skill ordering without any explicit supervision on transition structure, relying only on a sparse success reward.

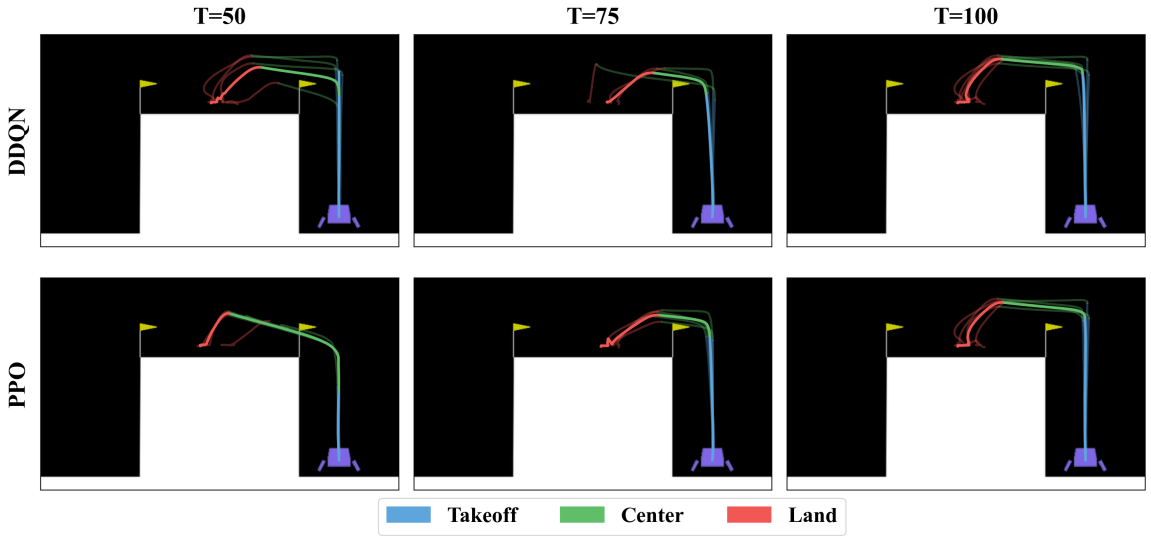


Figure 5.3: Learned skill sequencing for the LunarLander task. Each subplot shows the trajectory of the lander across evaluation episodes, colored by the active skill option: Takeoff (blue), Center (green) and Land (red). Columns correspond to the option duration $T \in \{50, 75, 100\}$ and rows correspond to selector algorithm (DDQN, PPO). Bold lines show the mean trajectory averaged across episodes across training seeds, while faint lines show individual training seed means. Both DDQN and PPO reliably learn coherent skill sequences across all option durations.

At $T_{\text{option}} = 100$, trajectories are compact across seeds, with the selector reliably invoking each skill exactly once. At shorter horizons, greater variation is visible across seeds as the selector resamples mid-execution, producing sequences such as T, C, C, C, L or T, T, C, C, L. This reflects the selector compensating for incomplete skill execution within the shorter option horizon and suggests that the VFS embedding provides a meaningful signal at each decision point regardless of where within a skill's execution the selector is queried.

Taking a closer look at the VFS embedding, Figure 5.4 shows how it evolves over the course of a representative training seed. V_{takeoff} (blue) rises sharply as the agent ascends early in the episode, peaking near timestep 40 before declining as the agent moves to the top of the environment boundary and stabilizes. V_{center} (green) begins at a lower value and rises as horizontal movement becomes viable, remaining elevated until the agent centers over the landing pad. V_{land} remains low during takeoff and

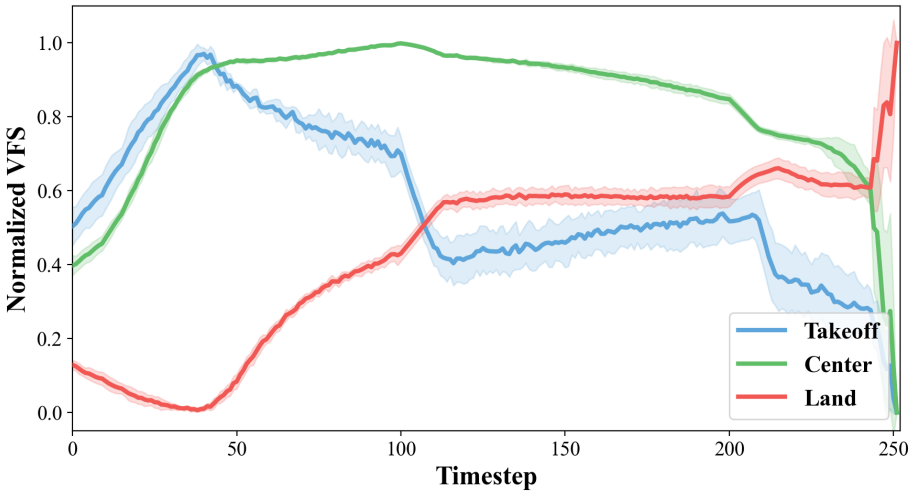


Figure 5.4: Normalized VFS values over a representative training seed, with mean and standard deviation computed across 5 evaluation episodes. Each value $V_k(s)$ is normalized to $[0, 1]$ within its own range to facilitate comparisons across skills. Values match the structure of the task: the takeoff critic peaks early as the lander gains altitude, the center critic remains elevated as the lander traverses horizontally, and the land critic rises once the lander is positioned over the target zone.

rises gradually as the lander approaches the pad, spiking sharply once positioned over the landing zone. This pattern confirms that the VFS embedding captures a task-relevant state abstraction, with each value function peaking in the states for which its corresponding skill becomes most executable and aligning with the learned sequences of Figure 5.3. We note that the hselector observes the raw VFS values rather than the normalized values shown here, which are presented solely for analysis.

5.2.3 Baseline Comparisons

Table 5.3 summarizes the performance of baselines against our approach, while Figure 5.5 illustrates baseline trajectory outcomes across success, truncation and termination categories. While all baselines fall short, ranging from 0 to 0.72 success rate, our approach achieves consistent success.

Flat PPO with a sparse reward fails entirely, achieving zero success across training seeds. Figure 5.5 reveals how, without any intermediate signal, the agent never

Table 5.3: Comparison of our approach against baselines on LunarLander. Total training steps for VFS include skill pretraining plus selector training. Success rate is defined as landing safely and is averaged across 5 seeds and 5 evaluation episodes per seed.

Approach	**Steps**	**Eval Success Rate**
VFS + DDQN Selector	2.3M	1.0
VFS + PPO Selector	2.3M	1.0
Flat PPO + Sparse Reward	5M	0
Flat PPO + Intermediate Goal Rewards	3M	0.4 ± 0.49
Flat PPO + Distance Reward	3M	0.72 ± 0.3
Flat PPO + Random Network Distillation	3M	0.48 ± 0.45
HIRO-PPO	5M	0.44 ± 0.49

discovers the landing behavior and instead terminates frequently. This underscores a fundamental task difficulty: landing is not merely a matter of reaching the correct (x, y) position but also requires landing *safely*, a combination of constraints that is unlikely to be stumbled upon by random exploration under a sparse reward.

Augmenting the reward with intermediate goals partially mitigates this, achieving an average success rate of 0.4, though with high variance across seeds (± 0.49). Examining Figure 5.5, truncation episodes show the agent consistently reaching the vicinity of the pad but failing to complete the landing, while termination episodes show it crashing upon approach. This pattern suggests that intermediate goals are effective at discovering the landing zone but still insufficient for landing safely.

RND achieves a moderate success rate of 0.48 (± 0.45), with a notably consistent hover pattern in truncation episodes wherein the agent discovers the pad region through exploration but struggles to execute the precise touchdown required for a safe landing. Similarly, termination episodes show an identification of the landing pad region but occasional crash landings.

The distance reward achieves the highest flat baseline success rate at 0.72 (± 0.30), with notably different failure characteristics than the other conditions. Failed episodes are almost exclusively terminations from landing with high velocity rather than timeouts. This reflects the nature of the reward: continuous proximity feedback produces

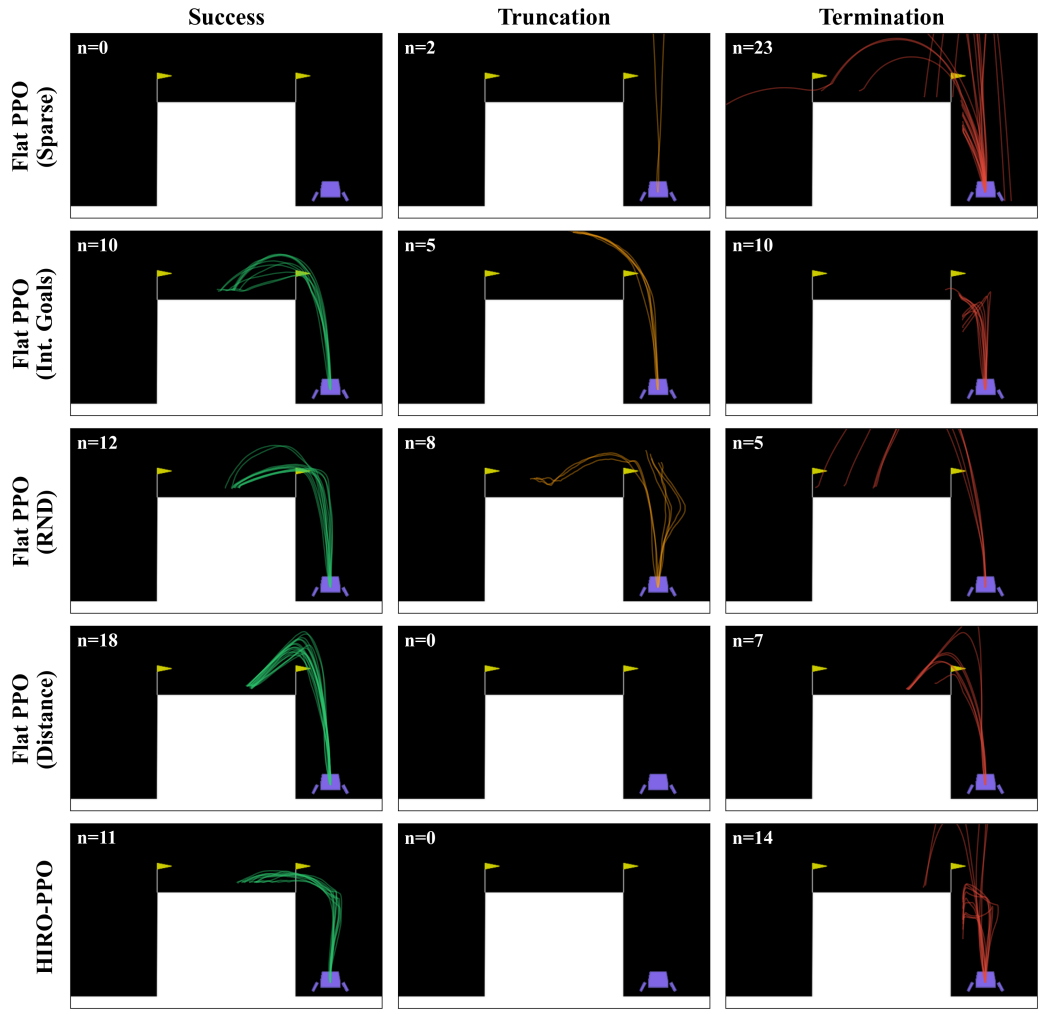


Figure 5.5: Trajectory outcomes for baselines on LunarLander, where each row corresponds to a baseline and each column to an outcome category: Success (green), Truncation (orange) and Termination (red). Because the variation in trajectories makes it difficult to average, we instead display individual evaluation episodes across all training seeds. n denotes the number of evaluation episodes in each category (25 total).

more directed trajectories but may still send the agent into the pad at incorrect angles or velocities, resulting in crash landings.

Finally, HIRO-PPO achieves a success rate of 0.44 ± 0.49, comparable to the flat PPO baselines. Examining the trajectories, successful episodes show the agent discovering a coherent liftoff and approach path, though at a noticeably lower altitude than other baselines. Termination episodes reveal a consistent failure mode of flying into the right boundary of the platform, as well as a less frequent failure mode of rising

too high and crashing onto the landing pad. This suggests that while the emergent goal-directed hierarchy discovers the landing zone, the emergent goal space does not reliably align with the task of landing, producing high variance across seeds.

Together, these results illustrate that neither reward engineering, intrinsic motivation nor emergent hierarchical structure reliably solve the task. In contrast, our approach achieves perfect success in only 2.3M total timesteps by first defining skills that render safe movement and landing tractable in isolation before composition.

5.3 Discussion

The LunarLander results provide a clean proof of concept for the VFS framework across several dimensions. All three skills achieve 100% success in isolation, confirming that the chosen decomposition renders each subproblem tractable. This is non-trivial: the landing skill in particular requires satisfying simultaneous velocity and contact constraints that the baselines often fail to discover. The downstream implication is that the quality of the skill decomposition is a primary determinant of overall performance. We note that it may be possible to achieve similar success with a coarser decomposition, such as by combining takeoff and center into a single skill. For our purposes, however, using three skills offered a more rigorous test of the selector.

The selector then recovers the correct skill ordering from a sparse task reward alone, without any explicit supervision on transition structure. This is possible because the VFS embedding evolves coherently with task progress, peaking for each skill at precisely the states where that skill is most executable. The fact that the selector exploits this structure consistently across both algorithms and all option horizons suggests that value functions of pretrained skills constitute a robust state abstraction for high-level decision-making, validating a core component of our framework.

Turning to the effect of T_{option}, the option horizon affects the structure of learned

sequences but not final performance. At shorter horizons, the selector produces longer sequences as it resamples mid-execution, while at $T = 100$ it reliably invokes each skill exactly once. That all T_{option} values converge to the same behavior again suggests the VFS embedding provides a consistent feasibility signal regardless of where within a skill's execution the selector is queried.

Turning to the selector algorithm, DDQN converges slightly faster than PPO in this setting, consistent with its suitability for small discrete action spaces. With only three options, ϵ-greedy exploration directly samples all skills within a few steps, ensuring diverse experience in the replay buffer even early in training. PPO's entropy-based exploration is softer and more dependent on the policy gradient signal to encourage option diversity, which may be a disadvantage when early rollouts are uninformative. The interaction between algorithm and option horizon becomes apparent at $T = 50$, where the option horizon is only one fourth of the skill training horizon. The selector is therefore frequently queried on partial skill executions that have not yet reached their subgoal states and may be less informative to the VFS embedding. The requirement of additional skill training at this horizon for PPO suggests that the option horizon and skill training budget interact, and that shorter horizons place greater demands on skill behavioral clarity and quality.

Finally, the baseline comparisons confirm that neither reward engineering, intrinsic motivation, nor emergent hierarchical structure reliably solves the task within the training budget. We qualify this conclusion on three grounds. First, the flat baselines were trained for 3M steps, which is within the range of our total pipeline cost, but some conditions show non-zero variable success that may consolidate with further training. Secondly, while we tuned key hyperparameters for each baseline, our search was non-exhaustive. Finally, HIRO-PPO faces an additional challenge in this environment, as LunarLander introduces boundary terminations that are not present in the continuous control settings for which HIRO was originally designed, potentially

making it more difficult for the low-level policy to learn stable goal-directed behavior before hitting boundaries. Taken together, however, the baselines still demonstrate a fundamental struggle to overcome the simultaneous challenges of discovering task structure and learning to execute within it, whereas our approach separates these problems entirely. Looking forward, the remaining question is whether our framework scales to higher-dimensional continuous control settings, where both the skill policies and VFS embedding are more complex.

Chapter 6

Bipedal Walker Experiment

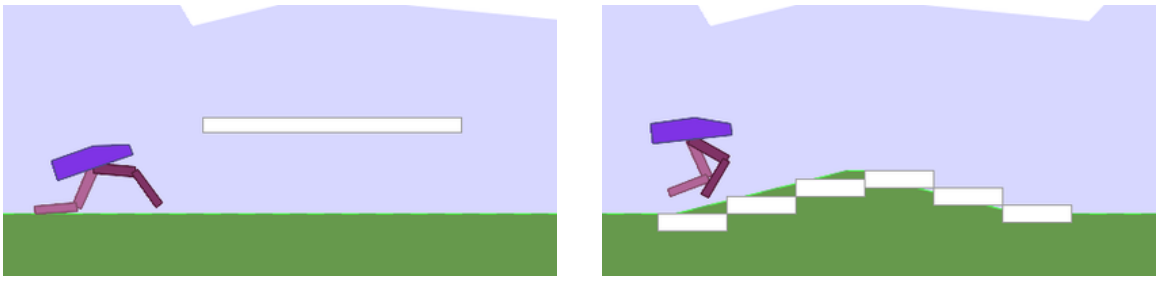
6.1 Methods

The BipedalWalker experiment extends the validation from LunarLander to a continuous control setting with a larger state space and task horizon. This experiment is designed to test whether our approach holds in a more demanding setting before we apply it to humanoid locomotion.

6.1.1 Environment

We use a modified version of the BipedalWalker environment [35], which features a bipedal robot with a 24-dimensional continuous state space including hull angle, angular velocity, joint angles, joint velocities and Light-Detection-And-Ranging (LIDAR) range measurements alongside a four-dimensional continuous action space controlling hip and knee torques. The original task requires the walker to traverse flat terrain, which a flat policy can solve without multi-stage behavior. To necessitate skill composition, we introduce two terrain variants that require qualitatively different locomotion strategies (Figure 6.1). The grass variant places a low overhang obstacle above flat terrain, requiring the walker to crouch and pass underneath. The stairs variant places

a staircase, requiring the walker to coordinate stepping behavior.



(a) Grass terrain with overhang obstacle (b) Stairs terrain with staircase

Figure 6.1: Modified BipedalWalker Environment

6.1.2 Skill Design and Training

We define two low-level skills corresponding to the two terrain variants: grass, which navigates the walker underneath the overhang obstacle while maintaining forward velocity, and stairs, which requires coordinated leg-lifting and single-leg balancing. Each skill is trained independently using PPO with a skill-specific reward function and terrain obstacle described below. Hyperparameters are provided in Appendix D.

Grass: The grass skill is initialized on flat terrain before an overhang obstacle. Its reward function augments the base environment reward with a bonus proportional to forward velocity, encouraging the walker to pass underneath the obstacle. A sparse success reward is issued when the walker clears the far edge of the obstacle.

Stairs: The stairs skill is initialized on flat terrain before a stairs obstacle. Its reward function augments the base environment reward with a bonus proportional to forward velocity and a penalty discouraging too wide of a hip joint angle to encourage upright posture. An additional success reward is issued upon completing the obstacle.

6.1.3 High-level Policy Training

Given the two trained skill policies, we train a high-level selector to choose which skill to execute at each decision point. The selector observes the VFS embedding

$Z(s) = [V_{\text{grass}}(s), V_{\text{stairs}}(s)]$, where each component is the value estimated by the corresponding skill's frozen critic. The selector chooses a skill, which then executes for a fixed number of environment steps T_{option} before the selector is queried again. The selector is evaluated in a sequential environment in which both obstacles appear, requiring the selector to select the correct skill at each stage using only the task reward and the structure implicit in $Z(s)$. The selector receives a sparse success reward upon completing the full sequence and a termination penalty for falling. We evaluate two selector algorithms—DDQN and PPO—with $T_{\text{option}} \in \{100, 200, 250\}$, reflecting the longer execution horizon required for BipedalWalker as compared to LunarLander.

6.1.4 Baselines and Evaluation

We compare against the flat PPO baselines described in Section 4.2, instantiated for the modified BipedalWalker environment. The sparse reward baseline receives a success signal only upon completing the full sequence. The intermediate goal reward baseline receives incremental rewards throughout the sequence, providing additional incentive to move forward, as well as a bonus for maintaining a closed hip angle when on the stairs. The distance reward baseline follows the original environment reward, but we append an additional reward for forward velocity and remove the penalty for excessive energy expenditure. The RND baseline augments the sparse reward with an intrinsic motivation bonus. Finally, due to time limitations, we do not compare to HIRO-PPO for this domain, as our primary goal is to test the scalability of our approach to continuous control rather than evaluate the quality of the skill decomposition compared to an emergent one. Baseline hyperparameters are provided in Appendix D.

We evaluate on task success rate, averaged across 5 training seeds and 5 evaluation episodes per seed, and report total environment training steps. For our approach, total steps include skill and selector training. Because BipedalWalker involves continuous

action spaces and longer episode horizons than LunarLander, this experiment provides a stronger test of whether our framework scales to more demanding settings.

6.2 Results

6.2.1 Skills

Figure 6.2 shows a filmstrip of each skill's learned behavior on a representative episode. The grass skill successfully learns to crouch and maintain forward velocity while passing underneath the overhang obstacle. The stairs skill learns to lift its legs and step upward and downward through the staircase. Noticeably, the two skills produce behaviorally distinct policies, with the grass skill adopting a wider hip angle and lower body position than the stairs. As shown in Table 6.1, both skills achieve 100% success averaged across 5 training seeds and 5 evaluation episodes per seed. Stairs require more training than grass, reflecting the greater coordination demand.

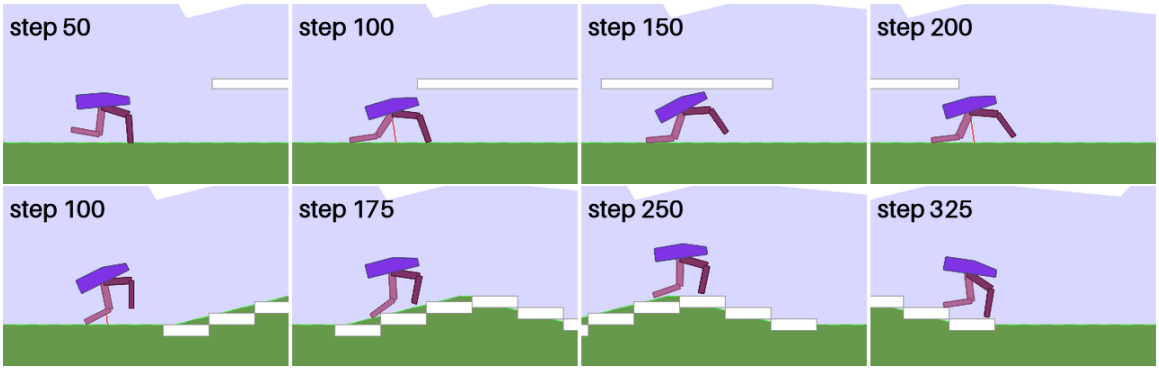


Figure 6.2: Filmstrips of learned skill policies on a representative episode in BipedalWalker. **Top:** Grass. **Bottom:** Stairs.

Table 6.1: BipedalWalker skills performance. Success rate, defined as successful obstacle completion, is averaged across 5 training seeds, each evaluated over 5 episodes.

Skill	Steps (Millions)	Evaluation Success Rate
Grass	1M	100%
Stairs	1.5M	100%

6.2.2 Selector

Table 6.2: BipedalWalker selector performance across algorithms and option horizons. Success rate, defined as successful completion of the obstacle sequence, is averaged across 5 training seeds, each evaluated over 5 episodes.

Algorithm	$T_{\textbf{option}}$	**Steps (Millions)**	**Evaluation Success Rate**
DDQN	100	1.1M	1.0 ± 0
DDQN	200	0.6M	0.92 ± 0.109
DDQN	250	0.6M	0.96 ± 0.089
PPO	100	1.1M	0.92 ± 0.109
PPO	200	0.6M	0.92 ± 0.109
PPO	250	0.5M	1.0 ± 0

$T = 100$ requires extended training time for the stairs skill by 0.5M steps, which we include in the training steps above.

Table 6.2 summarizes selector performance across algorithms and option horizons. Both DDQN and PPO achieve high success rates across conditions, with DDQN $T_{\text{option}} = 100$ and PPO $T_{\text{option}} = 250$ achieving perfect success (1.0 ± 0). The remaining conditions cluster between 0.92 and 0.95, indicating consistent but occasionally imperfect sequencing and skill execution. The shorter $T_{\text{option}} = 100$ requires extended stair training, as more frequent option selection means the stair skill is invoked from a broader range of states, demanding greater recovery ability from the low-level policy.

Figure 6.3 shows the skill selection heatmap across a fixed terrain, in which an overhang obstacle is followed by a stairs obstacle. The selector reliably applies the grass skill on the overhang segment and the stairs skill on the stairs segment, demonstrating clear terrain-skill correspondence. This correspondence is slightly asymmetric: the stairs skill is never selected on the overhang terrain, while grass is occasionally selected on the stairs terrain. This asymmetry is consistent with the structure of the skills themselves, as the upright stepping behavior of the stairs skill is entirely ill-suited to the low-clearance overhang, while the grass skill may occasionally be able to succeed on the stairs, particularly on the descent to flat terrain.

Figure 6.4 shows the normalized VFS values over a representative training seed.

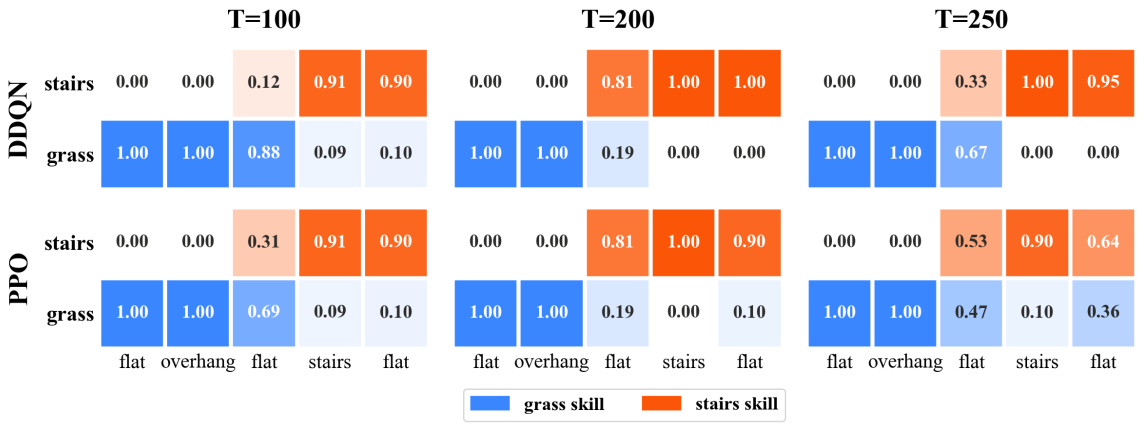


Figure 6.3: Skill selection heatmap against the sequential terrain for all algorithm and option horizon conditions. Each cell shows the fraction of timesteps the grass (blue) or stairs (orange) skill was selected on each terrain segment. The selector reliably applies the grass skill on the overhang segment and the stairs skill on the stairs segment, with occasional grass selection on the descending stairs portion.

V_{grass} begins elevated and remains high throughout the overhang segment before dropping sharply at approximately timestep 300, where the terrain transitions to stairs. V_{stairs} remains relatively stable during the flat and overhang phase before dipping briefly around timestep 300, then rising to its highest values over the stairs terrain. This dip reflects the agent's postural adjustment out of the crouched grass configuration: because the stairs critic was trained on upright stepping behavior, the transitional crouched posture is both outside its expected distribution and associated with lower reward, resulting in a lower value estimate. As the agent recovers its stepping gait, V_{stairs} rises and stabilizes, as visible in the transition filmstrip (Figure 6.5).

The greater noisiness of the VFS signal here relative to LunarLander is worth noting. In LunarLander, the observation space directly encodes global task progress through (x, y) position and velocity, and the reward structure of each skill encourages attaining a fixed target position and stable velocity. The resulting value functions therefore evolve smoothly with task phase, as previously shown in Figure 5.4. In BipedalWalker, by contrast, the observation space consists of joint angles, joint velocities, hull orientation and LIDAR range measurements. Furthermore, the skill

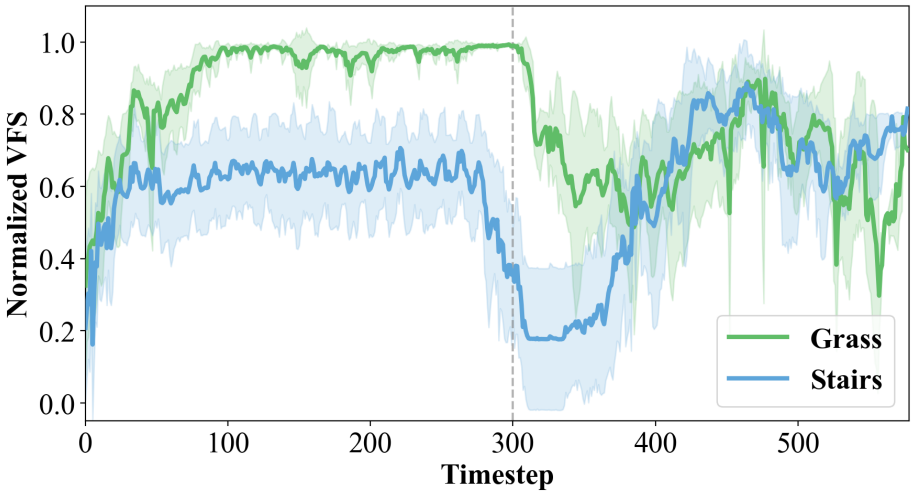


Figure 6.4: Normalized VFS values over a representative evaluation episode, with mean and standard deviation computed across five evaluation episodes. V_{grass} remains elevated throughout the overhang segment before dropping sharply at the terrain transition around timestep 300. V_{stairs} dips briefly at the transition before rising to its highest values over the stairs terrain.

Figure 6.5: Filmstrip of a sample skill transition at timestep 300 from Figure 6.4, with the corresponding V_{stairs} value annotated at each frame. The agent enters the stairs skill in a crouched posture (step 300, $V_{\text{stairs}} = 0.65$), adjusts through an intermediate configuration and recovers to an upright stepping gait by step 375 ($V_{\text{stairs}} = 0.90$).

reward functions reward proximity to target joint positions and successful completion of obstacles. The value functions therefore reflect both terrain context and the agent's current postural state simultaneously. This, combined with the fact that each skill's critic is trained only on its own terrain and thus encounters out of distribution states during sequencing, may contribute to the greater noisiness of the VFS signal.

However, the skills themselves are relatively well-differentiated, meaning that the selector can quickly learn from experience, for example, that executing the stairs skill on the overhang terrain leads to failure. Additionally, the selector learns an arbitrary

function over the VFS embedding, meaning that as long as the value functions produce consistent patterns given an observation, the selector can learn to exploit those patterns. Thus, the noisiness of the embedding reduces its transparency in this setting but does not necessarily undermine its utility as a basis for decision-making.

6.2.3 Baseline Comparisons

Table 6.3: Comparison of our approach against flat PPO baselines on Bipedal Walker. Total training steps for VFS include skill pretraining plus selector training. Mean sequence progress represents the fraction of the total obstacle sequence completed per episode, computed as the mean final x position divided by the total sequence length, averaged across 5 seeds and 5 evaluation episodes per seed.

Approach	**Steps**	**Mean Sequence Progress**
VFS + DDQN Selector ($T_{\text{option}} = 200$)	3.1M	0.95 ± 0.09
Sparse Reward	4M	0.13 ± 0.001
Intermediate Goal Rewards	4M	0.63 ± 0.26
Distance Reward	4M	0.82 ± 0.28
Random Network Distillation	4M	0.15 ± 0.02

Based on the consistent results across algorithms and option horizons reported in Table 6.2, we select DDQN with $T_{\text{option}} = 200$ as our representative configuration for baseline comparisons. Table 6.3 summarizes performance across all approaches. Our approach achieves a mean sequence progress of 0.95 ± 0.09, outperforming all flat baselines. In contrast, the sparse baseline nearly fails entirely at 0.13 ± 0.001, confirming that the task is too long-horizon for a sparse success reward alone to provide meaningful signal. As Figure 6.6 illustrates, the sparse baseline ultimately adopts a stationary position, choosing to avoid falling rather than progressing forward. The RND result is similar, with a mean sequence progress of 0.15 ± 0.02 and a similar stationary position reflected in its corresponding filmstrip.

The intermediate goals and distance baselines perform considerably better, with a mean sequence progress of 0.63 ± 0.26 and 0.82 ± 0.28 respectively. One notable difference between them is the speed at which they accomplish the sequence, as

demonstrated in Figure 6.6. The distance baseline, which is rewarded proportionally to forward movement, reaches the end of the staircase much earlier and therefore advances further before falling. In contrast, the intermediate goals baseline is only rewarded at intermediate waypoints, and it frequently fails due to truncation rather than termination, thus not progressing quite as far through the obstacle sequence.

Taking a closer look at the learned policies of the intermediate goals and distance baselines in Figure 6.6, we note that both adopt a wider hip angle when ascending the staircase, and the distinction between their body configurations on the different obstacles is overall less pronounced than in our approach, where each skill is trained to specialize in a particular terrain regime. This mirrors an observation from our selector results, where the grass skill occasionally succeeds on the stairs obstacle despite never being trained on it, suggesting that a sufficiently capable general locomotion policy may constitute a kind of universal motor primitive across terrain types. Taken together, these observations raise the possibility that our chosen skill decomposition, while beneficial within our training budget, may not be strictly necessary given sufficient flat training time and reward shaping.

6.3 Discussion

The BipedalWalker results extend the pattern observed in LunarLander to a more demanding continuous control setting. The core finding is consistent: our approach produces more reliable sequence completion than any flat baseline given the same training budget.

The failure of the RND baseline suggests that even intrinsic motivation cannot bridge the credit assignment gap imposed by the full obstacle sequence under a sparse reward. One potential explanation is the nature of the observation space, which contains no positional references and consists only of joint angles and LIDAR readings.

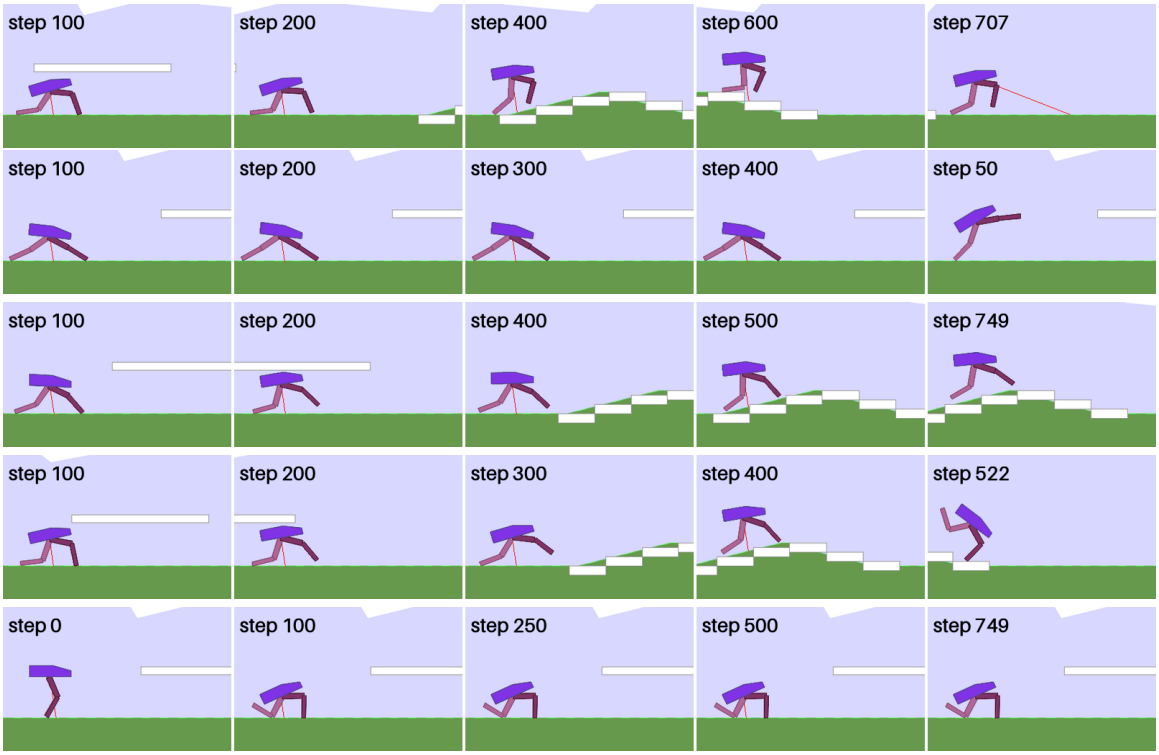


Figure 6.6: Representative filmstrip sequence comparisons between our approach and baselines. **Top to bottom:** Our approach (DDQN selector with $T_{\text{option}} = 200$, sparse baseline, intermediate goals baseline, distance baseline and RND baseline.

While RND may incentivize exploration of diverse body states and positions, those do not immediately translate to improved task performance, and indeed, many will lead to falls and incur a termination penalty. The long horizon of the task compounds this problem, as the agent must chain together many correct actions before any extrinsic reward is received, making it unlikely that intrinsic bonuses alone can guide meaningful progress. In this setting, only the distance and intermediate goal baselines achieve reasonable performance, suggesting that dense rewards are more useful for reducing the credit assignment burden in long-horizon continuous control tasks.

Looking ahead, one additional takeaway for the Humanoid experiments concerns the role of skill distinctiveness. In this experiment, the grass skill could occasionally succeed on the stairs obstacle despite not explicitly training on it, suggesting a certain degree of shared structure between the terrain skills. While this does not compromise our approach's performance in this setting, it does complicate the skill

selection decision that the selector has to solve and may have an increasingly large impact with a larger skill set size. In the Humanoid setting, however, we anticipate that this potential problem will be mediated by the highly distinct initial state distributions of each skill and increased complexity of the environment, which will likely make it more difficult for skills to generalize to unseen states. Having validated the core components of our framework in both LunarLander and BipedalWalker, we now turn to the Humanoid environment itself with these takeaways in mind.

Chapter 7

Humanoid Experiment

7.1 Methods

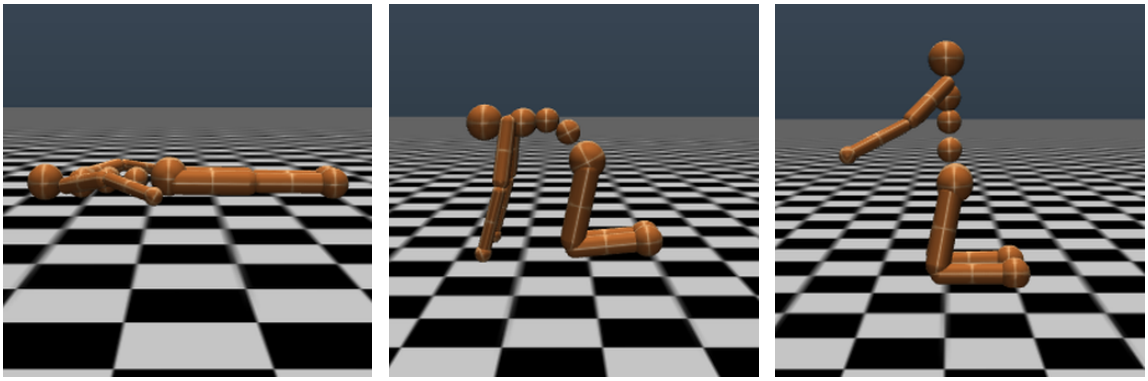
The Humanoid experiment represents the fullest instantiation of our approach and the central test of our hypothesis. Where LunarLander and BipedalWalker served as validation environments, the Humanoid skill decomposition is grounded directly in human motor development, with each skill corresponding to a postural transition that human infants reliably traverse on the path to upright locomotion. The high-dimensional state and action space also makes this a substantially more difficult exploration problem than the preceding environments.

7.1.1 Environment

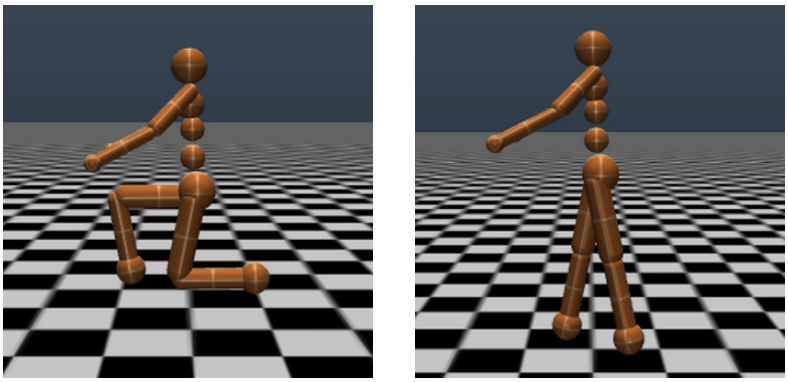
We use a modified version of the Humanoid-v4 environment [36] extended to support custom initial state distributions and reward functions. The humanoid model has a 376-dimensional observation space—including joint positions, velocities, center-of-mass inertia and velocities, actuator forces and external contact forces—and a 17-dimensional continuous action space controlling joint torques.

7.1.2 Skill Design and Training

We define five low-level skills corresponding to the five postural transitions of the developmental sequence: prone-to-crawl, crawl-to-kneel, kneel-to-lunge, lunge-to-stand and stand-to-walk. Because the humanoid state space is high-dimensional and the target postures are geometrically specific, each skill uses a goal-conditioned policy architecture in which the policy receives both the current observation and a fixed goal vector encoding the target joint configuration. Hyperparameters are provided in Appendix E.



(a) Prone-to-crawl (b) Crawl-to-kneel (c) Kneel-to-lunge



(d) Lunge-to-stand (e) Stand-to-walk

Figure 7.1: Initial State Distributions for Humanoid Skills

Initial State Distributions

The initial starting positions are illustrated in Figure 7.1. Each skill's target posture is the starting position of the subsequent skill in the sequence, in order to encourage that successful completion of one skill places the agent in a valid starting configuration for the next. To further improve robustness to imperfect skill handoffs during composition, noise is injected into the initial state distribution at the start of each episode, perturbing the relevant joints around the reference pose. Noise ranges were calibrated per skill by characterizing the terminal state distribution of each trained policy and setting the magnitude to cover the empirically observed variance at handoff.

Reward Functions

All skills share a common reward structure:

$$r = -5 \cdot \mathcal{L}_{\text{height}} - \mathcal{L}_{\text{joint}} - c_{vel} \cdot \mathcal{L}_{\text{qvel}} - 0.1 \cdot \mathcal{L}_{\text{ctrl}}$$

where $\mathcal{L}_{\text{height}} = |z - z^*|$ penalizes deviation from the target torso height, $\mathcal{L}_{\text{joint}} = |q - q^*|$ penalizes deviation from the target joint configuration, $\mathcal{L}_{\text{qvel}} = |q_{\text{vel}}|_2$ penalizes joint velocity to discourage erratic movement and $\mathcal{L}_{\text{u}} = |u|_2^2$ is the standard control cost penalizing joint torques. The joint error penalty operates over a task-relevant subset of joints, excluding arms and root orientation joints. For stand-to-walk, the joint velocity penalty is omitted, and a healthy reward $r_{\text{healthy}} = 5$ and forward velocity bonus $r_{\text{fwd}} = v_x + v_y$ are appended to encourage forward motion.

Skill Training

Individual skill training required substantial iterative tuning of reward weights, noise injection ranges and PPO hyperparameters. The primary challenge lies in balancing the height and joint configuration penalty against the joint velocity penalty. Achieving a target posture requires minimizing joint error while also limiting residual motion,

since the initial state distribution of each subsequent skill is defined not only over joint positions but also over velocities and contact forces. An overly strong velocity penalty, however, encourages the agent to remain stationary and results in sub-optimal joint configurations. Due to this sensitivity of skill training, we report results for the three seeds for which stable sets of skill policies were obtained.

The prone-to-crawl skill also provided particularly difficult to train reliably within our timestep budget. We therefore report results for the four-skill sequence beginning at crawl-to-kneel, and we treat prone-to-crawl as a separate preliminary skill result.

7.1.3 High-level Policy Training

Given the trained skill policies, we train a high-level selector to choose which skill to execute at each decision point. The selector observes the VFS embedding:

$$Z(s) = [V_{\text{crawl-to-kneel}}(s), V_{\text{kneel-to-lunge}}(s), V_{\text{lunge-to-stand}}(s), V_{\text{stand-to-walk}}(s)],$$

where each component is the value estimated by the corresponding skill's frozen critic. The selector chooses one of the four skills, which executes for a fixed number of environment steps T_{option} before the selector is queried again. We evaluate DDQN and PPO selectors with $T_{\text{option}} = 150$, as our noise calibration procedure limits the range of option horizons we can evaluate with a single set of skills. The selector is initialized from the crawl position and must coordinate the full skill sequence to achieve upright locomotion. It receives a reward and forward velocity bonus only when its torso height exceeds $z \geq 1.2$ meters.

7.1.4 Baselines and Evaluation

We compare against the baselines described in Section 4.2. For the flat PPO baselines, the sparse baseline receives the same indicator reward as our selector, namely a reward only when the torso height exceeds $z >= 1.2$ meters. The intermediate goals

baseline augments the observation with the joint error to each of four target sub-postures and provides incremental rewards upon reaching each target height, giving the agent additional directional signal without imposing hierarchical structure. The distance-based baseline uses the original environment reward, which provides a dense reward proportional to torso height ($\frac{z}{\Delta t}$). The RND baseline augments the sparse reward with an intrinsic exploration bonus to encourage coverage of the state space. Finally, HIRO-PPO represents our HRL baseline. As part of our hyperparameter tuning, we evaluate HIRO-PPO under the sparse, distance-based and intermediate goal reward formulations and report distance-based reward as the best-performing variant. Baseline hyperparameters are provided in Appendix E.

We evaluate our approach on two metrics: target height rate, or the fraction of episodes in which the agent achieves torso height $z \geq 1.2$m, and forward motion rate, or the fraction of episodes in which the agent proceeds to full bipedal locomotion. For baseline comparisons, we report average torso height per episode over training and torso height progression over evaluation episodes. We also assess movement quality through energy expenditure, center-of-mass stability and qualitative analysis of representative postural progressions. We define energy expenditure as the sum of squared torques applied to each joint per timestep, $\sum_i u_i^2$, measuring total control effort. We define the center-of-mass (COM) stability margin as the signed distance from the projected COM to the nearest edge of the convex hull of ground contact points. A positive COM stability margin indicates the COM lies within the support polygon and suggests postural stability, while a negative value indicates it has fallen outside the support base and suggests the agent is off-balance.

Finally, given the challenges described in Section 7.1.2, the results that follow should not be considered statistically significant. Skill performance is reported over three training seeds, each evaluated across five evaluation episodes. For our selector results, we train our high-level selector on three seeds per skill set and evaluate over

five evaluation episodes per seed. We then average across seeds within each skill set before computing the mean and standard deviation across the three skill sets. Baselines are also reported over three seeds, given their computational cost.

7.2 Results

7.2.1 Preliminary Prone-to-Crawl Skill

Table 7.1: Comparison of original prone-to-crawl skill versus decomposed prone-to-pushup and pushup-to-crawl skills. Terminal joint error is the distance between the agent's final joint configuration and the target pose, computed over twelve relevant joints. Terminal joint velocity is the L1 norm of all final joint velocities.

Skill	**Steps**	**Terminal Joint Error**	**Terminal Joint Velocity**
Prone-to-Crawl	8M	1.46 ± 0.57	50.58 ± 12.44
Prone-to-pushup	8M	0.37 ± 0.05	2.84 ± 0.90
Pushup-to-crawl	8M	0.51 ± 0.03	30.61 ± 6.79

The prone-to-crawl transition proved the most difficult to train reliably. A single policy attempting the full transition from a flat prone configuration to a crawl position achieved a terminal joint error of 1.46 ± 0.57 and terminal joint velocities of 50.58 ± 12.44, indicating both poor postural accuracy and high residual motion. We therefore experimented with decomposing this transition into two subskills—prone-to-pushup and push-up-to-crawl—where the first learns to lift the head and upper torso off the ground and the latter learns to reposition the limbs from this supported position into a crawl configuration, as demonstrated in Figure 7.2. This two-stage decomposition is inspired by the progression observed in human motor development, as pushing up onto the arms is a distinct milestone observed in human infant development [1, 6]. Despite convergence to more reasonable joint errors (0.37 ± 0.05 and 0.51 ± 0.03), the full two-step chain still lacks the robustness required for reliable composition, particularly with respect to the high joint velocities in the terminal state of pushup-

to-crawl. We therefore omit the prone-to-crawl transition from the composed sequence and report it as a preliminary result.

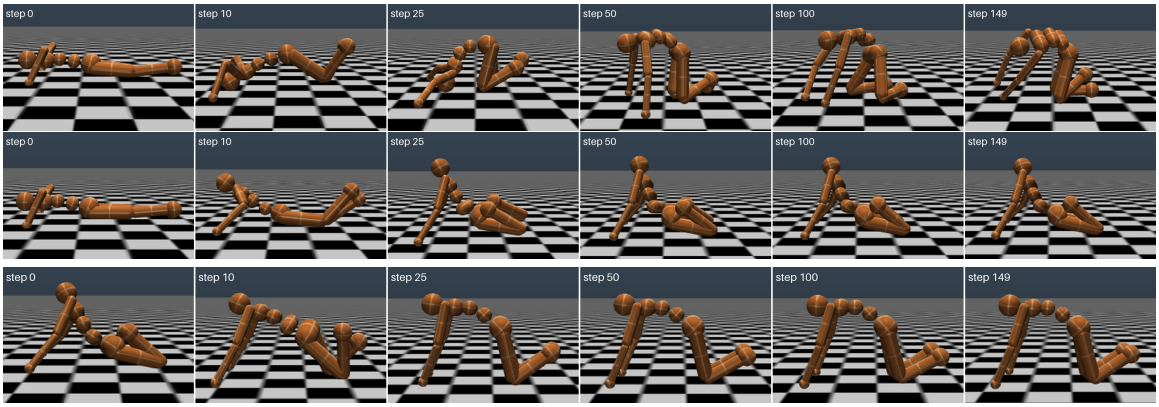


Figure 7.2: Representative filmstrips comparing the original prone-to-crawl skill (top) and the decomposed prone-to-pushup (middle) and pushup-to-crawl (bottom) skills.

7.2.2 Skills

Turning to the skills that make up our implemented developmental sequence, Table 7.2 summarizes training steps and terminal joint metrics across three training seeds. We note that some residual velocity is expected, particularly in postures with a higher center-of-mass where more balance is required, so a nonzero terminal velocity is not necessarily indicative of failure but rather greater balancing demands.

Table 7.2: Humanoid skill performance. Terminal joint error is the distance between the final joint positions and target joint positions, computed over twelve relevant joints. Terminal joint velocity is the L1 norm of all final joint velocities.

Skill	**Steps**	**Terminal Joint Error**	**Terminal Joint Velocity**
Crawl-to-kneel	10M	0.43 ± 0.09	0.99 ± 2.13
Kneel-to-lunge	10M	0.24 ± 0.03	4.99 ± 8.30
Lunge-to-stand	10M	0.39 ± 0.09	38.05 ± 14.5
Stand-to-walk	10M	0.42 ± 0.03	26.78 ± 4.41

Crawl-to-Kneel and Kneel-to-Lunge

To contextualize these metrics with a qualitative analysis, Figure 7.3 shows representative filmstrips of the crawl-to-kneel and kneel-to-lunge transitions. The crawl-to-kneel policy successfully lifts the torso from a four-limb support configuration into an upright kneeling posture, achieving a terminal joint error of 0.43 ± 0.09 and low terminal joint velocity (0.99 ± 2.13). The kneel-to-lunge policy achieves the lowest joint error of all skills (0.24 ± 0.03), though the high variance in terminal joint velocity reflects some residual motion in the extended leg.

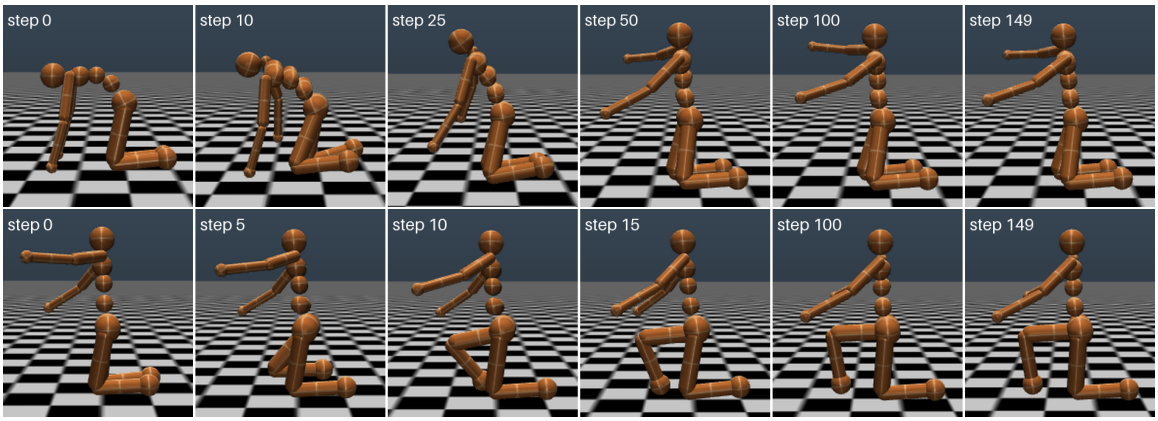


Figure 7.3: Representative filmstrips. **Top:** Crawl-to-kneel. **Bottom:** Kneel-to-lunge.

Lunge-to-Stand and Stand-to-Walk

Figure 7.4 shows the lunge-to-stand and stand-to-walk skills across a representative evaluation episode. The lunge-to-stand policy moves the agent to an upright position, adopting a slight split stance to maintain stability. While the terminal joint error is low (0.39 ± 0.09), the high terminal velocity (38.05 ± 14.5) indicates that the final standing posture carries residual momentum. As a result, we train stand-to-walk under substantially higher initial state noise than the preceding skills in order to make it robust to noisy handoffs produced by the lunge-to-stand skill. The resulting policy learns to stabilize from these perturbations–an example of which is shown in Figure 7.4, where the skill begins at a noticeable tilt with nonzero joint velocities.

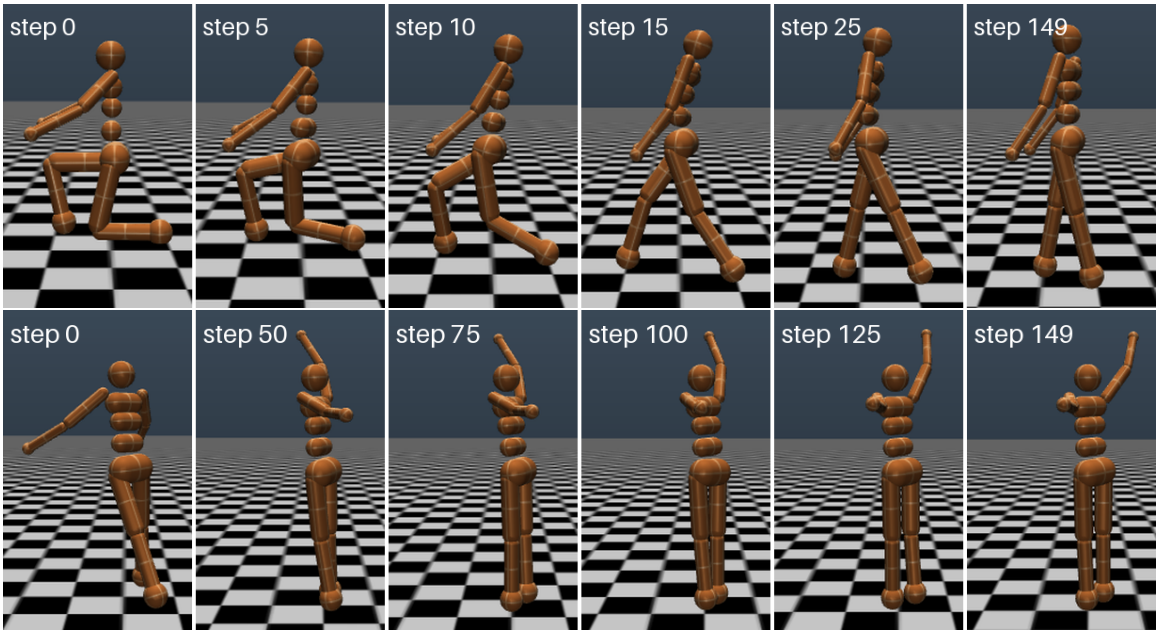


Figure 7.4: Representative filmstrips. **Top:** Lunge-to-stand. **Bottom:** Stand-to-walk, showcasing an example starting configuration after noise injection.

Taking a closer look at the quality of the resulting upright locomotion, we show foot contact statistics in Table 7.3, confirming a near-symmetric and alternating gait. The slight favoring of the left leg may be a result of the split stance, in which the left leg begins slightly forward. Furthermore, because we do not penalize upper limb position, the arms adopt an arbitrary configuration that best serves balance. Including a target arm position in the reward function would help reduce this slight visual unnaturalness, and we leave this as a straightforward extension.

Table 7.3: Gait statistics for stand-to-walk show near-regular bipedal locomotion.

Metric	**Left Foot**	**Right Foot**
Contact Fraction	0.49 ± 0.03	0.44 ± 0.02
Contact Duration	3.45 ± 0.06	3.04 ± 0.11
Number of Strides	25.6 ± 0.5	26.0 ± 0.6

Table 7.4: Humanoid selector performance with option duration $T_{\text{option}} = 150$. Steps denotes the number of selector training steps, excluding skill pretraining. Target Height Rate denotes the fraction of episodes in which the agent reaches the upright posture ($z \geq 1.2$m). Forward Motion Rate measures the fraction of successful episodes that successfully initiate forward motion. The gap between these metrics reflects quasi-failure cases where upright posture but not forward motion is attained.

Algorithm	$T_{\textbf{option}}$	**Steps**	**Target Height Rate**	**Forward Motion Rate**
DDQN	150	1.25M	0.933 ± 0.094	0.86 ± 0.02
PPO	150	2.5M	0.911 ± 0.12	0.81 ± 0.03

7.2.3 Selector

Table 7.4 summarizes the performance of the DDQN and PPO selectors, where we distinguish between two notions of success. The target height rate measures the fraction of episodes in which the agent reaches the upright posture (torso height $z \geq 1.2$m), reflecting the selector's ability to coordinate the postural transitions leading to standing. In contrast, the forward motion rate captures whether the full intended progression of skills from crawl-to-kneel through stand-to-walk is fully executed, thus initiating forward motion. The gap between these metrics reflects occurrences where successful upright standing is achieved but forward motion is not initiated.

Both approaches achieve high target height rates, with DDQN slightly outperforming PPO and converging in significantly less time. However, the slightly lower forward motion rates reveal occasional failures to transition from lunge-to-stand to stand-to-walk. Notably, the variance across seeds suggests that a single skill training seed accounts for the majority of these errors. In particular, this seed exhibits reduced robustness in the lunge-to-stand skill, leading to more episodes where the agent either falls or remains upright without initiating forward locomotion. Consequently, the observed performance differences appear to be driven more by variability in skill quality than by deficiencies in the high-level selector itself.

Taking a closer look at the selector behavior, Figure 7.5 plots skill activations as a function of torso height for both DDQN and PPO. On the left, solid curves

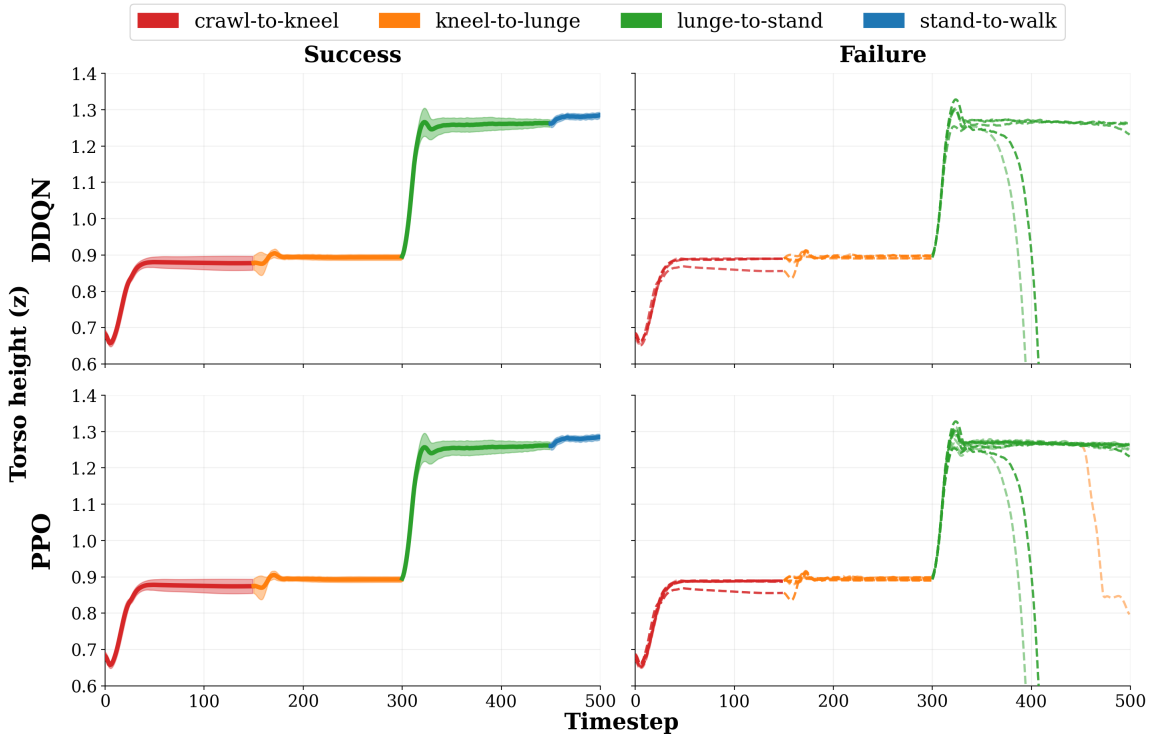


Figure 7.5: Torso height trajectories for DDQN (top) and PPO (bottom) selectors. **Left:** Solid lines show the mean torso height over successful evaluation episodes, with shaded regions denoting one standard deviation. **Right:** Dashed lines show individual trajectory failures, mainly due to lunge-to-stand failures. Colors indicate active skill.

represent the mean torso height over successful trajectories, where both selectors exhibit a clear and consistent progression through the developmental sequence. On the right, individual failed trajectories show cases where the developmental sequence is not fully executed. Here, we can clearly see that most failures arise from skill-level instabilities occurring mid skill-execution rather than errors in high-level sequencing. In particular, unsuccessful executions of the lunge-to-stand skill frequently lead to falls, as indicated by the dashed green trajectories that drop sharply.

A second failure mode occurs near the upright threshold and illustrates the cases in which the stand-to-walk skill is not executed (as captured by the forward motion rate in Table 7.4). Because both lunge-to-stand and stand-to-walk receive reward for achieving the target torso height, the selector may struggle to discriminate between them. Additionally, as noted in Section 7.2.2, the transition between lunge-to-stand

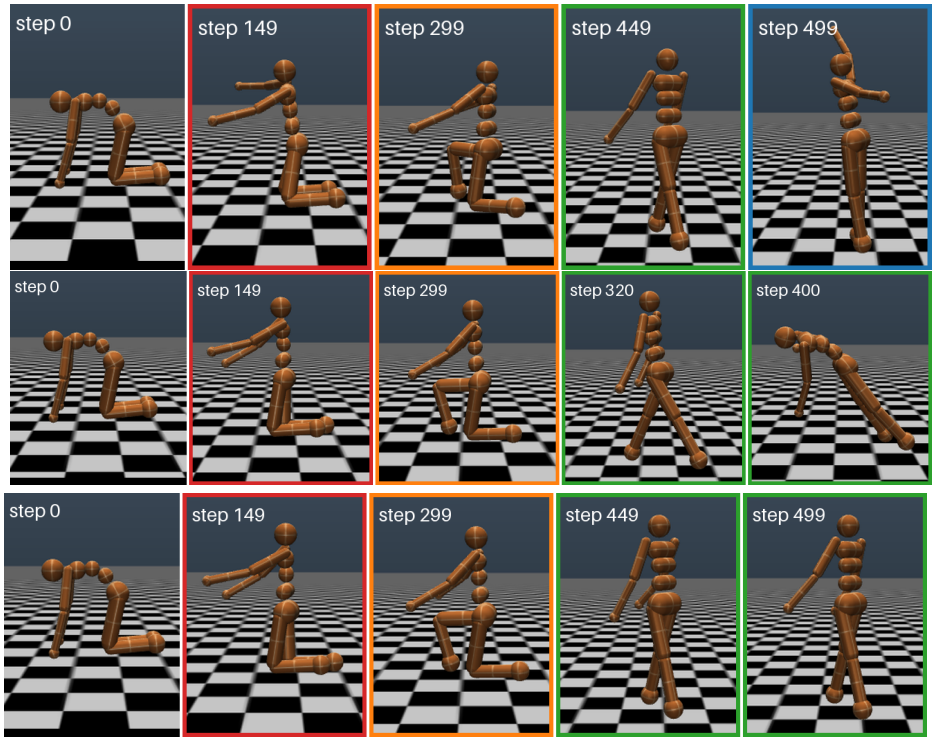


Figure 7.6: Representative filmstrips illustrating selector behavior. **Top:** Successful episode executing the full developmental sequence. **Middle:** Fall due to an unstable lunge-to-stand transition. **Bottom:** Non-fall failure that reaches the target height but does not initiate forward motion. Borders indicate active skill: crawl-to-kneel (red), kneel-to-lunge (orange), lunge-to-stand (green) and stand-to-walk (blue).

and stand-to-walk is particularly difficult and required injecting substantial noise into the initial state distribution of the latter. Thus, it is also possible that the selector occasionally avoids transitioning to stand-to-walk when it has learned that transition to be less stable. In other words, this quasi-failure mode, in which the target height is achieved but the stand-to-walk skill fails to execute, may also be related to the quality of skills rather than failures in the high-level selector.

Figure 7.6 provides qualitative examples of these behaviors. The top row shows a successful episode in which the selector executes the full developmental sequence and transitions to walking. The middle row illustrates a failure caused by an unstable lunge-to-stand transition that results in a fall. The bottom row depicts a non-fall failure in which the agent remains upright but continues executing lunge-to-stand

instead of transitioning to stand-to-walk and initiating forward motion.

Overall, these results suggest that the selector is effective at coordinating pretrained skills to achieve upright locomotion. The high target height rates confirm reliable sequencing of early postural transitions, while the gap between target height and forward motion rates highlights residual challenges in the lunge-to-stand to stand-to-walk handoff. This finding underscores the importance of stable and consistent skill policies for hierarchical control and suggests that further improvements in skill robustness are likely to yield corresponding gains in overall system performance.

7.2.4 Baseline Comparisons

To contextualize the performance of our proposed approach, we compare it against the flat and hierarchical baselines defined in Sections 4.2 and 7.1.4. We begin by comparing learning curves, for which we use average torso height per episode, given the differing reward functions across methods. We then compare the final learned policies of each baseline to our approach, considering torso height, center-of-mass stability and qualitative comparisons of postural progressions. We note that for the movement quality analysis, we show only the DDQN-based selector in order to generate cleaner comparisons within figures; however, both selector algorithms produce similar movement quality results given the shared underlying skills. We relegate energy expenditure evaluation to Appendix B.

Learning Dynamics

Figure 7.7 presents the learning curves for baseline methods over the first 40M steps of training on the left, while providing a zoomed-in view of the final 5M steps of training on the right. We place our approach, which we refer to as DevVFS-DDQN and DevVFS-PPO throughout this section, on the x-axis at 40M steps to account for the cost of skill pretraining and enable a fair comparison with baselines that are trained end-to-end. Our approach demonstrates rapid and stable convergence, quickly

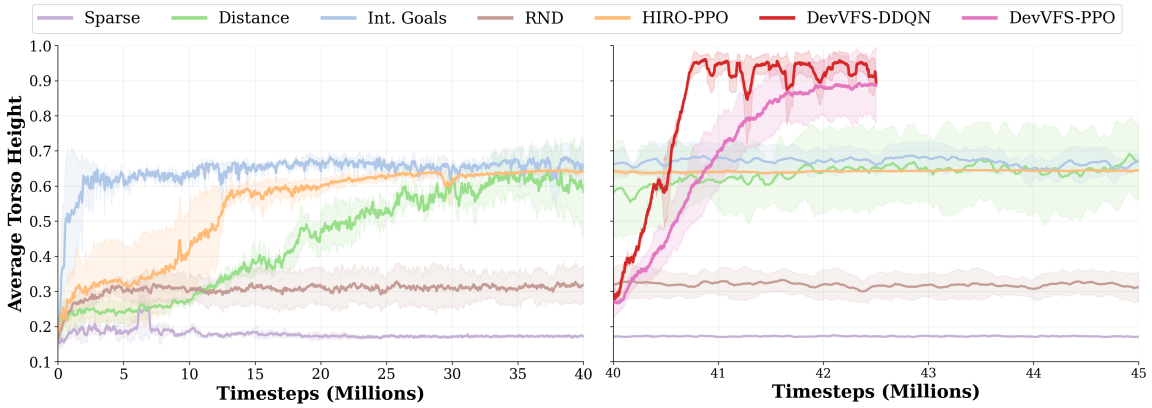


Figure 7.7: Average torso height per episode during training for all methods. **Left:** Training from 0-40M steps. **Right:** Final 5M training steps, with our approach appended at 40M steps to account for the cost of skill pretraining.

achieving and maintaining torso heights consistent with a fully upright posture. We note that due to the nature of our skills and the selected $T_{\text{option}} = 150$, our approach has an upper bound limit of average torso height near 1.0m, as it must progress through the developmental sequence and does not reach a standing posture until after halfway through the episode. Even despite this, we still see a clear learning advantage from our approach.

In contrast, the sparse and RND baselines plateau around 0.2 and 0.3m respectively. The distance based reward perform slightly better, achieving an average torso height near 0.6m by the end of training. The intermediate goals baseline performs noticeably better, achieving and maintaining an average torso height above 0.6m much earlier in training. Finally, HIRO-PPO initially learns relatively quickly as well but eventually plateaus around 0.6m and does not improve from there.

Sparse Baseline

As shown in Figure 7.8, the sparse reward baseline exhibits minimal progress toward upright posture, with torso height dropping almost immediately to around 0.2m during evaluation episodes, indicating a fall. This deficiency is further reflected in the COM stability margin, which frequently falls below zero, signaling poor balance until the agent falls completely to the ground. The representative filmstrip corroborates

these findings, showing the agent falling almost immediately before spending the rest of the episode on the ground, making little upward progress. Overall, this suggests that the task is not solvable for the sparse baseline, which lacks any structural prior.

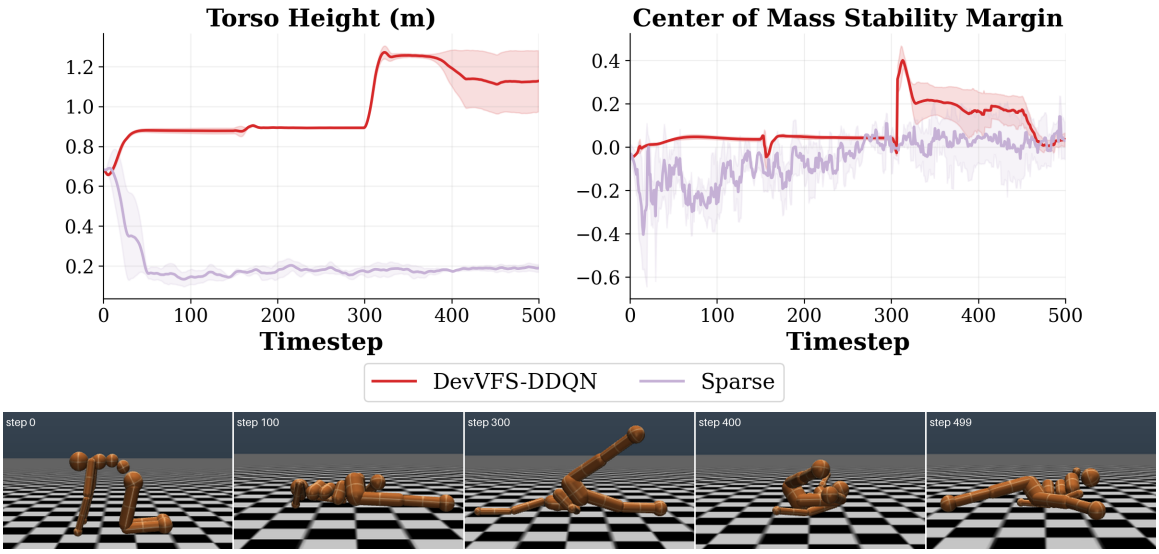


Figure 7.8: Comparison of sparse baseline and our approach over evaluation episodes. **Upper Left**: Torso height progression. **Upper Right**: Center of mass stability margin, where a positive value indicates a stable COM and a negative value indicates an unstable COM. **Bottom:** Representative baseline postural progression.

Intermediate Goals Baseline

As shown in Figure 7.9, the intermediate goals baseline demonstrates improved learning relative to the sparse setting, achieving evaluation torso heights in the range of 0.6-0.7m, though still failing to reach or sustain a fully upright stance. The COM stability margin notably remains above zero for most of the episode, indicating that the COM remains well-placed within the agent's base of support. Qualitatively, the representative filmstrip reveals that the agent progresses to a kneeling position, which enables it to achieve a slightly higher reward than in the initial crawl position. These results suggest that the intermediate goal signals provide some useful guidance in discovering the next milestone, namely the kneeling position. However, they remain insufficient alone to ensure the sequential coordination of the multiple transitions required for full upright locomotion, suggesting the presence of the two-level hierarchy

and temporal abstraction of skills are themselves useful.

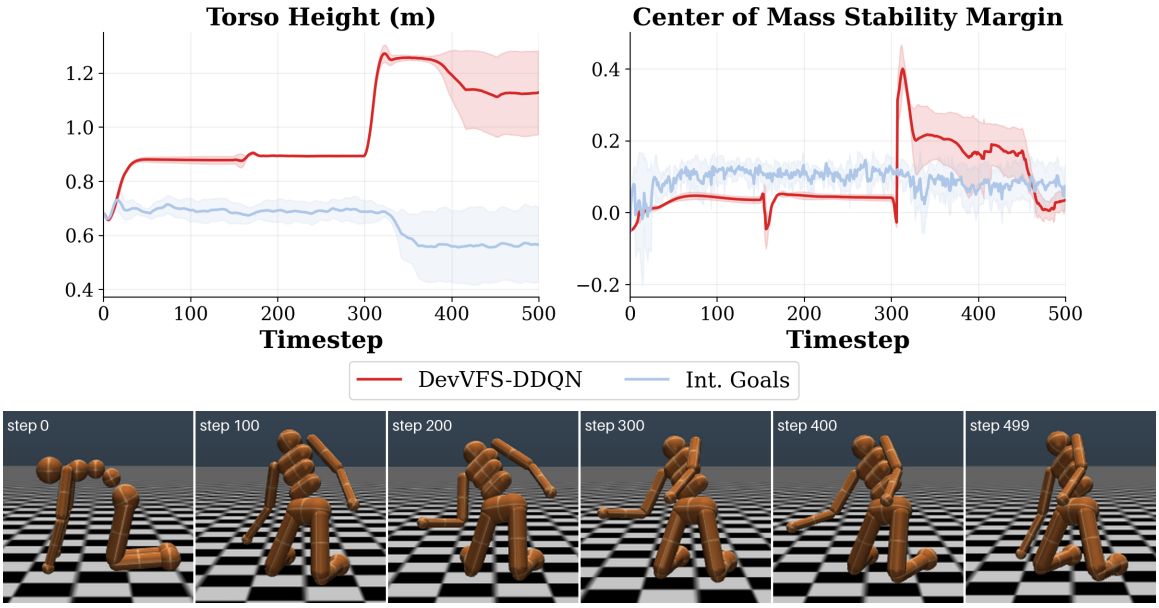


Figure 7.9: Comparison of intermediate goals baseline and our approach over evaluation episodes. **Upper Left**: Torso height progression. **Upper Right**: Center of mass stability margin, where a positive value indicates a stable COM and a negative value indicates an unstable COM. **Bottom:** Representative baseline postural progression.

Distance Baseline

As shown in Figure 7.10, the distance baseline achieves torso heights similar to the intermediate goals approach, though it fails to maintain that position for quite as long. The COM stability margin noticeably drops below zero towards the end of the episode, indicating more frequent and immediate falls. The representative filmstrip illustrates that while the agent's efforts are not coordinated into a fully upright position, suggesting that reward density alone cannot compete with our approach.

RND Baseline

The RND baseline encourages broader exploration and is the only flat baseline to occasionally achieve torso heights exceeding 1.0m during evaluation, as shown in Figure 7.11. However, these successes are transient and not well-sustained when compared to the stable performance of our approach. The COM stability margin is particularly telling, showcasing a significant drop below zero corresponding to the instability of

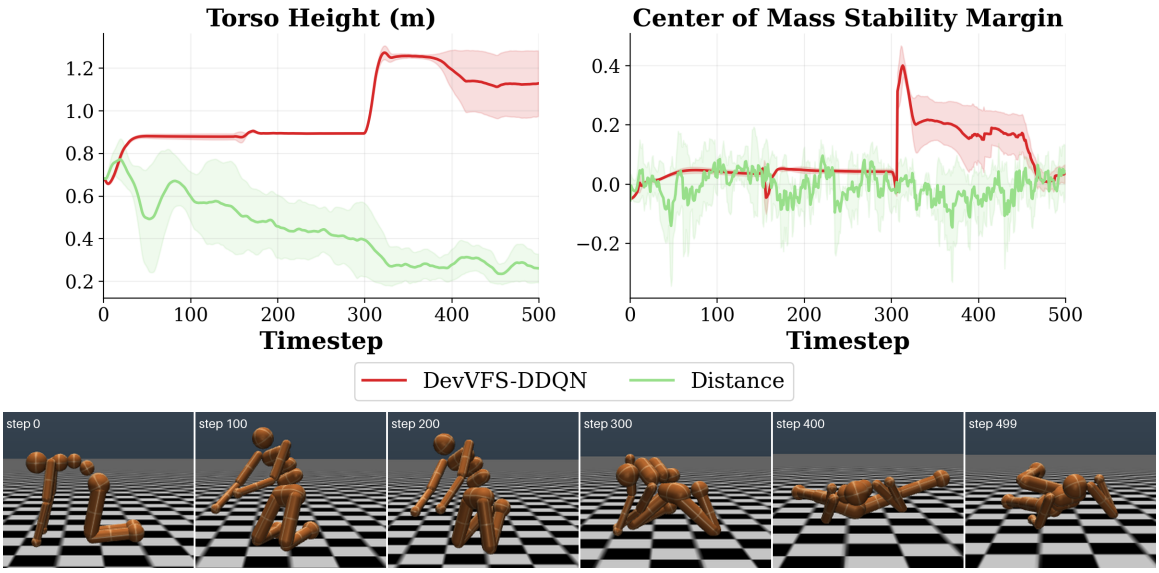


Figure 7.10: Comparison of distance baseline and our approach over evaluation episodes. **Upper Left**: Torso height progression. **Upper Right**: Center of mass stability margin, where a positive value indicates a stable COM and a negative value indicates an unstable COM. **Bottom:** Representative baseline postural progression.

its efforts to stand. The qualitative filmstrip further illustrates a rather precarious jump to a standing position early in training, followed by a near immediate fall and subsequent persistent efforts to lift the torso off the ground. We note that this standing position is not consistently achieved across all evaluation episodes, as indicated by the high variance of the torso height in Figure 7.11.

HIRO-PPO Baseline

Finally, the HIRO-PPO baseline achieves a uniquely distinct policy as compared to the other baselines. While it initially learns far faster than all but the intermediate goals baseline, its average torso height plateaus around 0.6m and remains consistent at evaluation time (Figure 7.7 and Figure 7.12). This suggests that it settles on a policy that achieves lower but consistent reward, rather than risking an upwards movement and falling to the ground. The COM stability index is slightly below zero but quite stable, suggesting a narrow but balanced base of support. The corresponding filmstrip corroborates these findings, showcasing a maintenance of the crawl position.

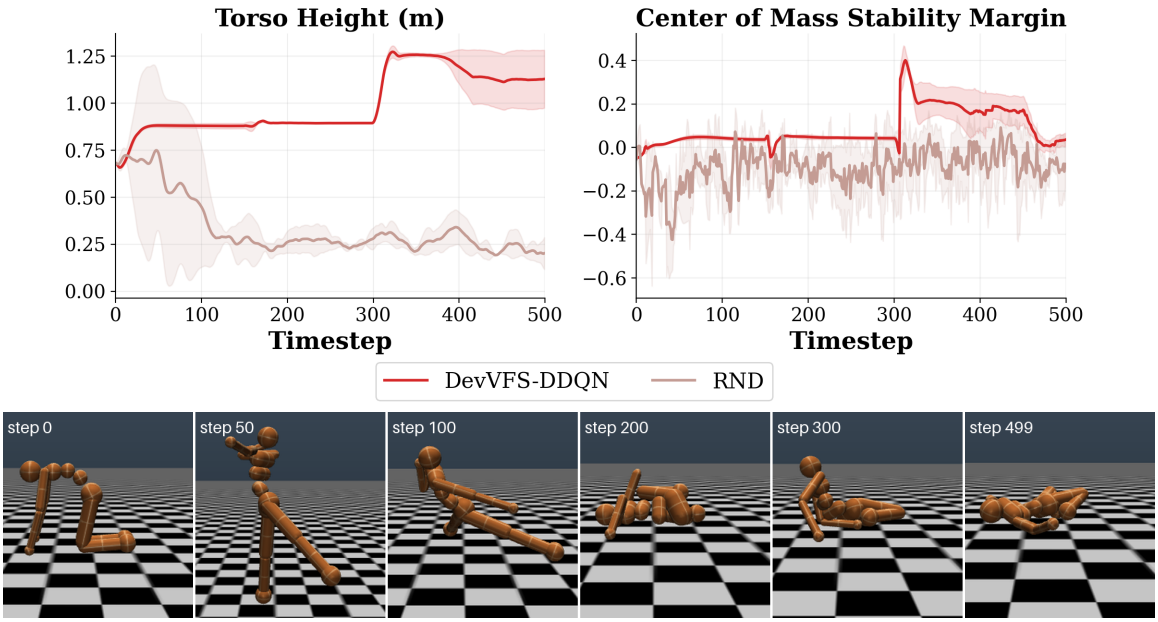


Figure 7.11: Comparison of RND baseline and our approach over evaluation episodes. **Upper Left**: Torso height progression. **Upper Right**: Center of mass stability margin, where a positive value indicates a stable COM and a negative value indicates an unstable COM. **Bottom:** Representative baseline postural progression.

We note two alternative explanations that might account for HIRO-PPO's inferior performance to our approach: insufficient exploration due to suboptimal hyperparameter choices and rewarding-hacking, wherein the reward given for maintaining a crawl position distracts from the potentially greater rewards of more upright postures. To address the possibility of the former, we conducted a comprehensive sweep across C, or the number of environment steps between high-level goal proposals, as well as the entropy coefficients for both the high- and low-level policies. To investigate the possibility of reward hacking, we also ran HIRO-PPO with the intermediate goal reward function and the sparse reward function. These experiments yielded comparable or worse outcomes, suggesting that the observed performance limitations are not a consequence of either case. The full results of these ablation studies are provided in Appendix B, and we discuss other explanations further in Section 7.3.

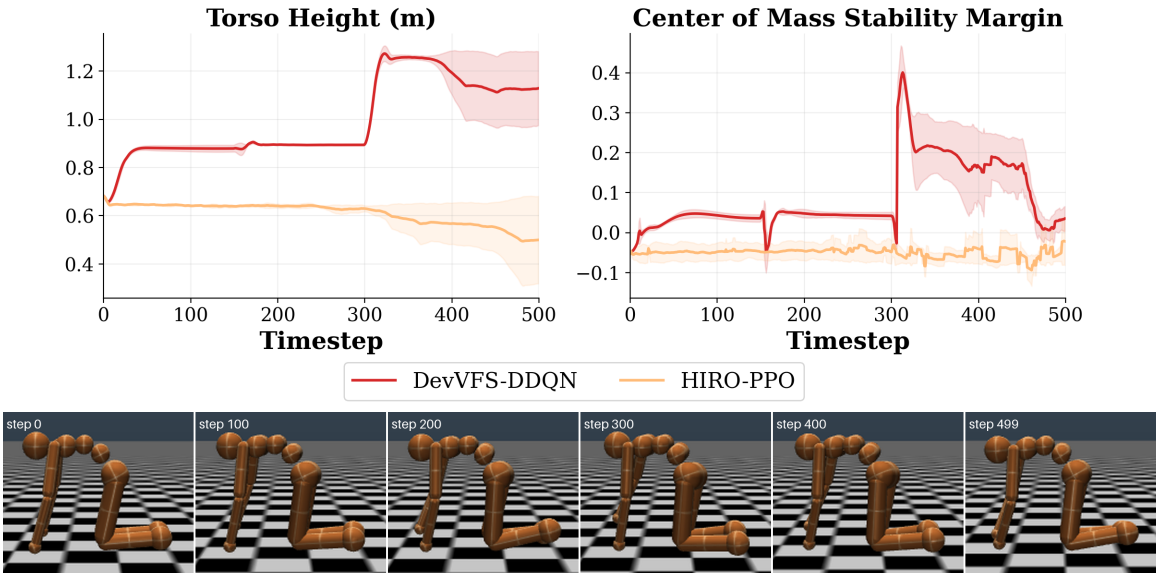


Figure 7.12: Comparison of HIRO-PPO baseline and our approach over evaluation episodes. **Upper Left**: Torso height progression. **Upper Right**: Center of mass stability margin, where a positive value indicates a stable COM and a negative value indicates an unstable COM. **Bottom:** Representative baseline postural progression.

7.3 Discussion

The Humanoid experiments demonstrate that our developmental hierarchy enables reliable coordination of complex postural transitions leading to upright locomotion. Our approach consistently achieves and maintains a stable standing and walking posture, significantly outperforming both flat and hierarchical baselines with respect to evaluation torso height, center-of-mass stability and qualitative postural progression analysis. Our approach also achieves this performance gain within the same range of total environment steps. These results support our hypothesis that leveraging the developmental sequence as a structural prior can substantially reduce the exploration burden and improve both performance and sample complexity for the task of learning to stand up and walk.

The Success of the Developmental Sequence

A key insight form our results is the effectiveness of the developmental sequence as a principled task decomposition. Compared to all baselines, the imposed progression

from crawl-to-kneel through stand-to-walk provides a structured pathway that guides both learning and execution. Because each transition corresponds to a biomechanically meaningful and locally stable subproblem, low-level policies can specialize in well-defined control regimes. In turn, the selector is afforded a set of skills that bias its exploration toward meaningful subgoals, and it only need learn to compose a much short sequence of temporally extended skills rather than several hundred steps worth of primitive actions. With the help of the VFS embedding, the high-level selector thus operates over a reduced and semantically meaningful decision space.

Interestingly, insights from human motor development proved useful even during difficulties with the prone-to-crawl transition. We initially defined this transition as moving directly from prone to crawl, grouping multiple prone strategies exhibited by humans into a single transition. However, human motor development specifically identifies the milestone of pushing up with arms as distinct from the task of moving to a crawl position [6, 1]. When we decomposed the prone-to-crawl transition into two corresponding skills—prone-to-pushup and pushup-to-crawl—the ability to learn the overall transition dramatically improved. This further reinforces that the developmental milestones carve the problem at natural discontinuities, defining subproblems that are not only meaningful to humans but tractable for learning agents. More broadly, these findings reinforce the notion that leveraging domain knowledge to define skill boundaries can yield decompositions that are more aligned with the underlying biomechanics of the task than those discovered purely through interaction.

The Challenge of Skill Chaining

While individual skills perform reliably in isolation, our experiments highlight the persistent challenge of chaining these skills into a seamless behavioral sequence. Failure cases in our approach are predominantly attributable to instabilities at skill boundaries, particularly surrounding the lunge-to-stand skill. This is reflected in the gap between the target height rate and the forward motion rate from Table 7.4, where

agents occasionally achieved an upright posture but failed to initiate locomotion.

These findings underscore a broader and well-established challenge in HRL: skill-chaining, wherein the terminal state distribution of one skill must fall within the viable initiation set of the next [43]. In continuous domains, this overlap is not guaranteed and must be actively engineered, as small discrepancies in termination states can place the agent outside the initiation region of the subsequent skill entirely, causing the chain to fail even when each skill is reliable in isolation. Our calibration of initial-state noise ranges to the empirical terminal distributions at each skill boundary is an attempt to address precisely this: by expanding each skill's effective initiation set to cover the variance introduced by its predecessor, we reduce the probability of handoff failure at composition time. However, this problem is further complicated by variability across training seeds, as the terminal state distributions themselves differ across runs. Noise ranges therefore need to be well-calibrated to the average termination behavior across seeds, and even then it may fail to fully cover the tails of this distribution.

Beyond discrepancies in terminal joint positions, skill composition is further sensitive to residual joint velocities and contact dynamics at skill boundaries. The lunge-to-stand transition illustrates this most clearly: while the terminal joint error is low for both this skill and the preceding kneel-to-lunge skill, the high and variable terminal joint velocities compound across the transition boundary. In turn, this occasionally places subsequent skills in the sequence in states that, although positionally within their initiation set, carry residual momentum that destabilize the transition. While our primary skill reward signal—the distance between the current positional configuration and target positional configuration—offers a dense and well-defined objective for achieving the desired posture, the appropriate treatment of joint velocities is less straightforward. Excessive regularization can discourage the exploratory movements necessary to reach the target configuration, whereas insufficient regularization results

in residual momentum that compromises the stability of subsequent skill transitions. Future work could further refine the velocity regularization weights and calibration of initial-state noise ranges to the empirical terminal distribution at each boundary.

Comparison to An Emergent Hierarchy

Turning to the comparison with our baselines, we first consider the performance of an emergent hierarchical structure, as represented by HIRO-PPO, relative to our imposed developmental hierarchical structure. Although HIRO-PPO initially demonstrates faster learning than most flat baselines, its performance plateaus at an intermediate torso height near the crawl position and fails to achieve upright locomotion. This suggests the learned policy settles on a conservative strategy, maintaining a low but stable posture rather than attempting a more challenging upward transition.

A key factor underlying this outcome is the reliance of HIRO on emergent subgoals proposed by the high-level policy, which must be executed by a learned low-level policy. When these subgoals are difficult to achieve or inconsistently realized, the high-level policy experiences increased uncertainty, wherein it is difficult to distinguish whether a proposed subgoal was poorly chosen or whether the low-level policy simply failed to execute it correctly. One plausible explanation for HIRO-PPO's behavior, then, is that the high-level policy converges to risk-averse behaviors by proposing more easily achievable subgoals that avoid failure, leading to conservative strategies that preserve stability over upward movement. A second contributing factor is the inherent difficulty of jointly optimizing the high- and low-level policies, as instability in one level can impede the other's ability to discover effective strategies.

In contrast, our approach equips the high-level policy with a set of pretrained skills designed to reliably achieve specific and biomechanically grounded objectives. This allows the high-level policy to make decisions with greater confidence in the expected outcome of each skill selection. This advantage is particularly pronounced in the Humanoid domain, where each skill is tailored to a specific initial state distribution and

incorrect skill selection typically results in immediate failure, enabling clearer credit assignment for the high-level policy. Collectively, these findings suggest that our imposed developmental hierarchy provides more reliable and effective coordination than an emergent hierarchical structure, enabling consistent progression through complex postural transitions toward upright locomotion.

We note that, while HIRO has been successfully implemented under various continuous control MuJoCo environments—including Ant Maze, Ant Push and Ant Fall—to our knowledge it has not been previously evaluated in the Humanoid domain. One remaining limitation of our work concerns our decision to reimplement HIRO using an on-policy PPO formulation. This choice was motivated by our desire to provide an "apples-to-apples" comparison with our skill training framework, and PPO is widely used and well-suited for continuous control. Nevertheless, this choice removes one potential benefit of the original HIRO algorithm: its off-policy replay buffer. In settings where successful transitions are sparse, replay allows those rare experiences to be reused, potentially facilitating more effective learning. This represents a potentially important deviation from the original HIRO formulation, and future work could reimplement the original off-policy HIRO algorithm as an additional baseline.

Comparison to Alternative Reward and Exploration Mechanisms

Turning to the flat PPO baselines, our results suggest that none of the individual formulations are alone sufficient to solve the task. The sparse baseline demonstrates that, without any structural guidance, the task is effectively intractable due to the difficulty of discovering long-horizon postural transitions without reward signals or more sophisticated exploration mechanisms. The distance baseline provides denser feedback to encourage incremental progress toward upright posture, yet it lacks the sequential structure suggesting plausible intermediate waypoints, resulting in only slight progress toward upright locomotion.

The intermediate goals baseline offers additional structure by incorporating milestone-

based signals corresponding to the developmental subpostures. While this enables the agent to more reliably achieve a relatively stable kneeling position, a single monolithic policy struggles to progress through multiple transitions sequentially. This suggests that, although intermediate goal information provides some useful context, the absence of the two-level hierarchy and specialized, temporally abstracted skills limits the agent's ability to coordinate complex, multi-stage behavior.

Finally, as the only flat method that occasionally achieves an upright posture during evaluation, the RND baseline underscores the importance of exploration. These successes are transient and reflect the undirected nature of RND's intrinsic motivation: the agent learns which spaces of the state space are valuable under a sparse reward, but it does not learn a policy robust enough to consistently transition through to those upright postures.

In comparison to the flat baselines, our approach can be understood as incorporating a form of directed exploration. Rather than searching broadly over the state space (RND), coordinating multiple transitions within a single monolithic policy (intermediate goals) or following a dense but structurally neutral gradient (distance), the developmental hierarchy channels learning toward task-relevant regions by decomposing the problem into tractable, biomechanically meaningful subproblems. Combined with the VFS representation, which offers a task-relevant state abstraction for the high-level policy decision-making, our approach overcomes the fundamental failure to coordinate complex behavior shared by all monolithic policies.

Summary

Taken together, these results demonstrate that our approach can substantially reduce the exploration burden of complex locomotion learning, enabling reliable upright locomotion where flat and emergent hierarchical baselines fail. The primary remaining bottleneck lies in skill robustness at transition boundaries and suggests that further refinement of skills may provide additional performance gains.

Chapter 8

Conclusions and Future Work

This thesis investigated whether structural priors derived from human motor development could reduce the exploration burden of hierarchical reinforcement learning for humanoid locomotion. We instantiated this hypothesis through the options and Value Function Spaces frameworks, imposing a fixed developmental skill sequence and constructing a compact, skill-centric state abstraction for high-level decision making. Validated first in LunarLander and BipedalWalker, where the framework achieves near-perfect task success across selector algorithms and option horizons, our approach scales to Humanoid. It reliably coordinates the postural sequence from crawl to upright locomotion, substantially outperforming both flat and HIRO-PPO baselines across torso height progression and movement quality evaluations.

Our work focuses specifically on the problem of getting up and walking, for which the developmental sequence offers a principled decomposition. We note, however, that the broader strategy of instantiating a known task decomposition as a fixed option set within an HRL framework is not specific to locomotion. Any domain in which a principled decomposition can be identified from prior knowledge is in principle amenable to this approach, and extending this framework to other such domains is one natural direction for future work.

Several concrete directions also remain within the Humanoid setting itself. The most immediate concerns skill chaining robustness, which is the primary bottleneck on overall system performance. Here, we suggest two pathways for future work. The first continues to pull on existing levers: more systematic reward engineering, tuning of velocity regularization weights and calibration of initial state noise injection ranges against empirical terminal state distributions at each skill boundary. The second pursues a more data-driven treatment of skill transitions altogether based on the observation that each skill in the developmental sequence induces a characteristic termination distribution over states. Future work could exploit these distributions directly, for instance by learning termination functions from the empirical state distributions at skill boundaries rather than relying on handcrafted noise pertubations.

The prone-to-crawl transition represents the other outstanding thread. While the two-stage decomposition into prone-to-pushup and pushup-to-crawl demonstrated meaningful progress, the high terminal joint velocities at skill boundaries prevented reliable integration into the composed sequence. Bringing this transition to the robustness standard of the remaining skills–through reward refinement, extended training or the termination-distribution approach above—would complete the full developmental hierarchy from the ground to bipedal locomotion.

Finally, two directions would strengthen the empirical picture. First, expanding the set of HRL baselines—including a full off-policy reimplementation of HIRO and additional hierarchical methods—would more thoroughly characterize the advantages and limitations of an imposed versus emergent decomposition. Second, an ablation isolating the contribution of the VFS representation, such as by comparing our selector with access to the full raw state against the VFS embedding, would directly assess how much of the performance gain is attributable to the state abstraction versus the skill decomposition alone. Together, these experiments would sharpen the conclusions of this thesis and clarity the relative contributions of each component.

Appendix A

Baseline Implementations

HIRO-PPO. The high-level policy proposes a goal $g_t \in \mathbb{R}^d$ every C environment steps, representing a desired displacement in the agent's observation space [10]. The low-level policy conditions on (s_t, g_t) and receives intrinsic reward

$$r_t^{\text{lo}} = -\|s_t[:d] + g_t - s_{t+1}[:d]\|_2,$$

which measures how closely the transition matches the proposed displacement. Between high-level proposals, the goal is updated via the transition function

$$h(s_t, g_t, s_{t+1}) = s_t[:d] + g_t - s_{t+1}[:d],$$

which keeps the absolute target fixed while re-expressing it as a new displacement from the current state. The high-level policy receives extrinsic reward only. Both levels are trained with PPO on-policy, updated once per rollout length.

RND. A fixed, randomly-initialized target network $f : \mathcal{S} \to \mathbb{R}^k$ maps observations to embeddings, and a learned predictor network $\hat{f}$ is trained to match it [37]. The intrinsic reward at each step is

$$r_t^{\text{int}} = \|\hat{f}(s_{t+1}) - f(s_{t+1})\|^2,$$

which is large for novel states the predictor has not yet seen and decays as the agent revisits familiar regions. A separate intrinsic value head and non-episodic intrinsic return are maintained alongside the standard extrinsic PPO update, with the combined advantage weighted by intrinsic and extrinsic coefficients. Observations are normalized before being passed to the RND networks to stabilize training.

Appendix B

Additional Humanoid Results

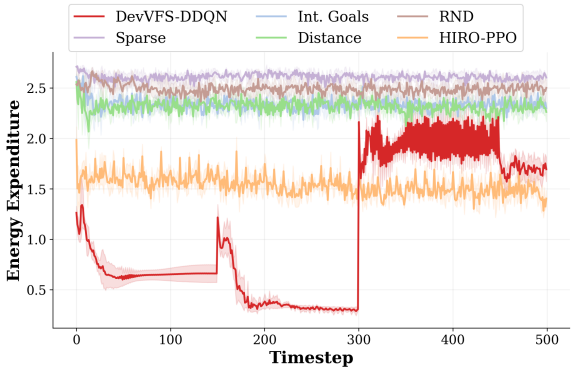


Figure B.1: Comparison of Energy Expenditure Across All Methods.

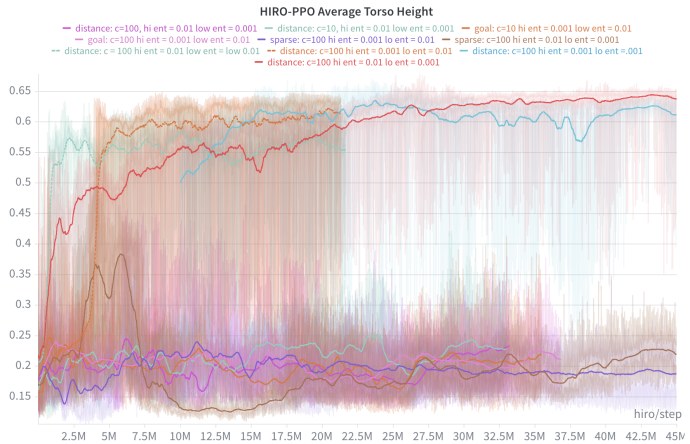


Figure B.2: HIRO-PPO Hyperparameter and Reward Formulation Tuning. We select the best performing trial, colored in red.

Appendix C

LunarLander Hyperparameters

Table C.1: LunarLander PPO skill hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [32, 32], Tanh	[16, 32, 64]
Actor learning rate	3e-4	[1e-3, 3e-4, 1e-4]
Critic learning rate	1e-3	[1e-3, 1e-4]
Rollout length	1024	[256, 512, 1024]
PPO epochs per update	5	Default
Clip parameter ϵ	0.2	Default
Discount factor γ	0.999	[0.99, 0.995, 0.999]
GAE parameter λ	0.98	[0.95, 0.98]
Entropy coefficient	0.0001	[0.01, 0.001, 0.0001]
Max grad norm	1.0	[0.5, 1.0, 5.0]

Table C.2: LunarLander PPO selector hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [16, 16], Tanh	Default
Actor learning rate	1e-3	[1e-3, 3e-4, 1e-4]
Critic learning rate	1e-4	[1e-3, 1e-4]
Rollout length	128	[64, 128, 256, 512]
PPO epochs per update	5	[5, 10]
Clip parameter ϵ	0.2	Default
Discount factor γ	0.999	Default
GAE parameter λ	0.95	Default
Entropy coefficient	0.01	[0.01, 0.001]
Max grad norm	1.0	[0.5, 1.0, 5.0]

Table C.3: LunarLander DDQN selector hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [16, 16], ReLU	Default
Learning rate	1e-3	[1e-3, 3e-3, 1e-4]
Batch size	256	[64, 128, 256, 512]
Replay buffer size	2000	Default
Discount factor γ	0.999	[0.99, 0.995, 0.999]
Updates per option	2	[1, 2]
ε-start	1.0	Default
ε-end	0.0	Default
ε-decay fraction	0.5	[0.5, 0.75]

Table C.4: LunarLander PPO hyperparameters shared across PPO baselines.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [16, 16], Tanh	Default
Actor learning rate	3e-4	[1e-3, 3e-4, 1e-4]
Critic learning rate	1e-3	[1e-3, 1e-4]
Rollout length	1024	[256, 512, 1024]
PPO epochs per update	5	Default
Clip parameter ϵ	0.2	Default
Discount factor γ	0.999	Default
GAE parameter λ	0.98	Default
Entropy coefficient	{1e-3, 1e-4}	[1e-2, 1e-3, 1e-4]
Max grad norm	1.0	Default

Table C.5: LunarLander RND-specific hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
RND embedding dimension	256	[16, 32, 128, 256]
RND target hidden size	256	[64, 256]
RND predictor hidden size	32	[16, 32, 64]
RND learning rate	1e-4	Default
Predictor update fraction	0.25	[0.1, 0.25]
Intrinsic coefficient	1.0	[1, 5, 10]
Extrinsic coefficient	1.0	[1, 2]

Table C.6: LunarLander HIRO-PPO-specific hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Low-level network architecture	MLP [32, 32], Tanh	Default
High-level network architecture	MLP [16, 16], Tanh	Default
Rollout length	2048	[1024, 2048, 4096]
Entropy coefficient	1e-4	[1e-2, 1e-3, 1e-4]
High-level re-proposal period C	100	[10, 50, 100]

Appendix D

BipedalWalker Hyperparameters

Table D.1: BipedalWalker PPO skill hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [64, 64], Tanh	[32, 64]
Actor learning rate	3e-4	[1e-4, 3e-4, 1e-3]
Critic learning rate	3e-3	[1e-4, 3e-3, 1e-3]
Rollout length	256	[128, 256, 512, 1024]
PPO epochs per update	5	Default
Clip parameter ϵ	0.2	Default
Discount factor γ	0.99	Default
GAE parameter λ	0.95	Default
Entropy coefficient	0.01	[0.01, 0.001]
Max grad norm	1.0	Default

Table D.2: BipedalWalker PPO selector hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [16, 16], Tanh	Default
Actor learning rate	1e-3	[1e-3, 3e-4, 1e-4]
Critic learning rate	1e-4	[1e-3, 1e-4]
Rollout length	128	[64, 128, 256]
PPO epochs per update	5	Default
Clip parameter ϵ	0.2	Default
Discount factor γ	0.999	Default
GAE parameter λ	0.95	Default
Entropy coefficient	0.01	[0.01, 0.001]
Max grad norm	1.0	Default

Table D.3: BipedalWalker DDQN selector hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [16, 16], ReLU	Default
Learning rate	1e-3	[1e-3, 1e-4]
Batch size	128	[64, 128, 256]
Replay buffer size	2000	Default
Discount factor γ	0.999	[0.99, 0.995, 0.999]
Updates per option	2	[1,2]
ε-start	1.0	Default
ε-end	0.0	Default
ε-decay fraction	0.75	[0.5, 0.75]
Target network sync period	1000	Default

Table D.4: BipedalWalker PPO hyperparameters shared across PPO baselines.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [64, 64], Tanh	[32, 64]
Actor learning rate	3e-4	[1e-4, 3e-4, 1e-3]
Critic learning rate	3e-3	[1e-4, 3e-3, 1e-3]
Rollout length	256	[128, 256, 512, 1024]
PPO epochs per update	5	Default
Clip parameter ϵ	0.2	Default
Discount factor γ	0.99	Default
GAE parameter λ	0.95	Default
Entropy coefficient	0.01	[0.01, 0.001]
Max grad norm	1.0	Default

Table D.5: BipedalWalker RND-specific hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
RND embedding dimension	256	Default
RND predictor hidden size	256	Default
RND target hidden size	256	Default
RND learning rate	1e-4	Default
RND update fraction	0.25	Default
Intrinsic reward coefficient	10.0	[1, 5, 10]
Extrinsic reward coefficient	1.0	[1, 2]

Appendix E

Humanoid Hyperparameters

Table E.1: Humanoid PPO skill hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture (base)	MLP [256], Tanh	Default
Network architecture (goal)	MLP [32], Tanh	Default
Actor learning rate	1e-4	[1e-3, 1e-4, 3e-4]
Critic learning rate	1e-3	[1e-3, 1e-4]
Rollout length	1024	[64, 256, 512, 1024]
PPO epochs per update	5	Default
Clip parameter ϵ	0.2	Default
Discount factor γ	0.99	[0.99, 0.995]
GAE parameter λ	0.95	Default
Entropy coefficient	1e-3	[1e-2, 1e-3]
Max grad norm	1.0	[0.5, 1.0, 5.0]

Table E.2: Humanoid PPO selector hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [16, 16], Tanh	Default
Actor learning rate	1e-3	Default
Critic learning rate	1e-4	Default
Rollout length	128	[64, 128, 256]
PPO epochs per update	5	Default
Clip parameter ϵ	0.2	Default
Discount factor γ	0.999	[0.99, 0.999]
GAE parameter λ	0.95	Default
Entropy coefficient	0.01	[0.01, 0.001]
Max grad norm	1.0	Default

Table E.3: Humanoid DDQN selector hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [16, 16], ReLU	Default
Learning rate	1e-3	Default
Batch size	128	[64, 128, 256]
Replay buffer size	2000	Default
Discount factor γ	0.999	[0.99, 0.999]
Updates per option	1	[1,2]
ε-start	1.0	Default
ε-end	0.0	Default
ε-decay fraction	0.75	[0.5, 0.75]
Target network sync period	1000	Default

Table E.4: Humanoid PPO hyperparameters shared across PPO baselines.

Hyperparameter	**Value**	**Tuned / Default**
Network architecture	MLP [256, 256], Tanh	Default
Actor learning rate	1e-4	Default
Critic learning rate	1e-3	Default
Rollout length	1024	[256, 1024]
PPO epochs per update	5	Default
Clip parameter ϵ	0.2	Default
Discount factor γ	0.99	Default
GAE parameter λ	0.95	Default
Entropy coefficient	0.001	[0.001, 0.01]
Max grad norm	1.0	Default

Table E.5: Humanoid RND-specific hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
RND embedding dimension	256	Default
RND target hidden size	128	Default
RND predictor hidden size	128	Default
RND learning rate	1e-4	Default
Predictor update fraction	0.25	Default
Intrinsic coefficient	10	[1, 5, 10]
Extrinsic coefficient	1.0	[1, 2]

Table E.6: Humanoid HIRO-PPO-specific hyperparameters.

Hyperparameter	**Value**	**Tuned / Default**
Low-level network architecture	MLP [256, 256], Tanh	Default
High-level network architecture	MLP [256, 256], Tanh	Default
Low-level entropy coefficient	1e-3	[1e-2, 1e-3]
High-level entropy coefficient	1e-2	[1e-2, 1e-3]
Rollout length	2048	[1024, 2048, 4096]
High-level re-proposal period c	100	[10, 50, 100]

Bibliography

[1] K. E. Adolph and J. M. Franchak, "The Development of Motor Behavior," *Wiley Interdisciplinary Reviews: Cognitive Science*, vol. 8, no. 1-2, p. e1430, 2017.

[2] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, T. P. Lillicrap, and M. A. Riedmiller, "DeepMind Control Suite," *CoRR*, 2018. [Online]. Available: http://arxiv.org/abs/1801.00690

[3] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," 2018. [Online]. Available: https://arxiv.org/abs/1801.01290

[4] X. He, R. Dong, Z. Chen, and S. Gupta, "Learning Getting-Up Policies for Real-World Humanoid Robots," 2025. [Online]. Available: https://arxiv.org/abs/2502.12152

[5] T. Tao, M. Wilson, R. Gou, and M. van de Panne, "Learning to Get Up," ACM SIGGRAPH 2022 Conference Proceedings, 2022.

[6] K. E. Adolph and S. R. Robinson, "The Road to Walking: What Learning to Walk Tells Us About Development," in *The Oxford Handbook of Developmental Psychology, Vol. 1: Body and Mind*, P. D. Zelazo, Ed. Oxford University Press, 2013.

[7] K. E. Adolph, "Learning to Move," *Current Directions in Psychological Science*, vol. 17, no. 3, pp. 213–218, 2008.

[8] A. G. Barto and S. Mahadevan, "Recent Advances in Hierarchical Reinforcement Learning," *Discrete Event Dynamic Systems*, vol. 13, pp. 41–77, 2003.

[9] R. S. Sutton, D. Precup, and S. Singh, "Between MDPs and semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning," *Artificial Intelligence*, vol. 112, no. 1–2, pp. 181–211, 1999.

[10] O. Nachum, S. Gu, H. Lee, and S. Levine, "Data-Efficient Hierarchical Reinforcement Learning," 2018. [Online]. Available: https://arxiv.org/abs/1805.08296

[11] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu, "FeUdal Networks for Hierarchical Reinforcement Learning," 2017. [Online]. Available: https://arxiv.org/abs/1703.01161

[12] P.-L. Bacon, J. Harb, and D. Precup, "The Option-Critic Architecture," 2016. [Online]. Available: https://arxiv.org/abs/1609.05140

[13] D. Shah, P. Xu, Y. Lu, T. Xiao, A. Toshev, S. Levine, and B. Ichter, "Value Function Spaces: Skill-Centric State Abstractions for Long-Horizon Reasoning," 2022. [Online]. Available: https://arxiv.org/abs/2111.03189

[14] R. Bellman, "A Markovian Decision Process," *Journal of Mathematics and Mechanics*, pp. 679–684, 1957.

[15] M. L. Puterman, *Markov Decision Processes—Discrete Stochastic Dynamic Programming*. New York: John Wiley & Sons, 1994.

[16] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, pp. 279–292, 1992.

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing Atari with Deep Reinforcement Learning," *arXiv preprint arXiv:1312.5602*, 2013. [Online]. Available: https://arxiv.org/abs/1312.5602

[18] H. van Hasselt, "Double Q-learning," in *Advances in Neural Information Processing Systems*, vol. 23, 2010.

[19] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-learning," 2015. [Online]. Available: https://arxiv.org/abs/1509.06461

[20] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation," in *Advances in Neural Information Processing Systems*, vol. 12, 1999.

[21] L. C. Baird III, "Advantage Updating," Wright Laboratory, Wright-Patterson Air Force Base, Tech. Rep., 1993.

[22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[23] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-Dimensional Continuous Control Using Generalized Advantage Estimation," 2015. [Online]. Available: https://arxiv.org/abs/1506.02438

[24] P. Dayan and G. E. Hinton, "Feudal Reinforcement Learning," *Advances in Neural Information Processing Systems, vol. 5*, 1992.

[25] T. G. Dietterich, "Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition," 1999. [Online]. Available: https://arxiv.org/abs/cs/9905014

[26] R. Parr and S. Russell, "Reinforcement Learning with Hierarchies of Machines," in *Advances in Neural Information Processing Systems*, vol. 10. MIT Press, 1997.

[27] N. K. Jong, T. Hester, and P. Stone, "The Utility of Temporal Abstraction in Reinforcement Learning," in *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*. International Foundation for Autonomous Agents and Multiagent Systems, 2008, pp. 299–306.

[28] G. Konidaris, "On the Necessity of Abstraction," *Current Opinion in Behavioral Sciences*, vol. 29, pp. 1–7, 2019.

[29] A. Solway, C. Diuk, N. Córdova, D. Yee, A. G. Barto, Y. Niv, and M. M. Botvinick, "Optimal Behavioral Hierarchy," *PLOS Computational Biology*, vol. 10, no. 8, p. e1003779, 2014.

[30] K. E. Adolph, S. E. Berger, and A. J. Leo, "Developmental Continuity? Crawling, Cruising, and Walking," *Developmental Science*, vol. 14, no. 2, pp. 306–318, 2011.

[31] O. Atun-Einy, S. E. Berger, and A. Scher, "Pulling to Stand: Common Trajectories and Individual Differences in Development," *Developmental Psychobiology*, vol. 54, no. 2, pp. 187–198, 2012.

[32] S. L. Thurman and R. Rose, "Infants' Organization of Pull-to-Stand Behaviors During Play: A Longitudinal Investigation," *Infant Behavior and Development*, vol. 78, p. 102033, 2025.

[33] A. Harutyunyan, W. Dabney, D. Borsa, N. Heess, R. Munos, and D. Precup, "The Termination Critic," 2019. [Online]. Available: https://arxiv.org/abs/1902.09996

[34] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz,

B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng, "Do As I Can, Not As I Say: Grounding Language in Robotic Affordances," 2022. [Online]. Available: https://arxiv.org/abs/2204.01691

[35] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI Gym," 2016. [Online]. Available: http://arxiv.org/abs/1606.01540

[36] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A Physics Engine for Model-Based Control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2012, pp. 5026–5033.

[37] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, "Exploration by Random Network Distillation," *arXiv preprint arXiv:1810.12894*, 2018. [Online]. Available: https://arxiv.org/abs/1810.12894

[38] A. Y. Ng, D. Harada, and S. J. Russell, "Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping," in *Proceedings of the Sixteenth International Conference on Machine Learning*, 1999, pp. 278–287.

[39] N. Chentanez, A. G. Barto, and S. P. Singh, "Intrinsically Motivated Reinforcement Learning," in *Advances in Neural Information Processing Systems*, vol. 17, 2004.

[40] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous Control with Deep Reinforcement Learning," 2019. [Online]. Available: https://arxiv.org/abs/1509.02971

[41] P. Henderson, W.-D. Chang, P.-L. Bacon, D. Meger, J. Pineau, and D. Precup, "OptionGAN: Learning Joint Reward-Policy Options using Generative Adversarial Inverse Reinforcement Learning," 2017. [Online]. Available: https://arxiv.org/abs/1709.06683

[42] S. Kakade, "On the Sample Complexity of Reinforcement Learning," Ph.D. dissertation, University College London, 2003.

[43] G. Konidaris and A. G. Barto, "Skill Discovery in Continuous Reinforcement Learning Domains using Skill Chaining," in *Advances in Neural Information Processing Systems*, vol. 22, 2009.