International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-14 Issue-11, October 2025

A Minecraft Agent Based on a Hierarchical Deep Reinforcement Learning Model

Arjun Panwar

Abstract: ***Open-world games such as Minecraft pose significant challenges for reinforcement learning (RL) systems due to their long-horizon objectives, sparse rewards, and requirement for compositional skill learning. This study investigates how a Hierarchical Deep Reinforcement Learning (HDRL) approach can improve agent performance and sample efficiency in such complex environments. We develop a hierarchical agent composed of three interconnected levels: (i) a high-level planner responsible for decomposing tasks into subtasks using the options framework for temporal abstraction, (ii) mid-level controllers that manage reusable subtasks such as resource gathering, crafting, and smelting, and (iii) a low-level visuomotor policy that interacts with the environment through human-like keyboard and mouse inputs. The agent's learning pipeline integrates pretraining from human demonstration datasets (MineRL) and large-scale video pretraining (VPT) to establish behavioural priors before reinforcement learning fine-tuning. This design leverages modern hierarchical algorithms such as Option-Critic, FeUdal Networks (FuN), HIRO, and Hierarchical Actor-Critic (HAC), enabling the agent to operate across multiple temporal scales. Evaluation is conducted using Obtain Diamond-style benchmarks and BASALT "reward-free" tasks to measure generalization and human alignment. Ablation studies assess the effect of each hierarchical layer, the inclusion of demonstrations, and large-scale video-based priors on overall performance. Results indicate that HDRL substantially enhances task completion rates and sample efficiency compared to monolithic RL agents, particularly in long-horizon and reward-sparse scenarios. This research was conducted to address the limitations of existing RL systems in complex, open-ended worlds and to explore how hierarchical structures can bridge the gap between low-level control and high-level planning. The findings demonstrate that hierarchical reinforcement learning provides a scalable and interpretable framework for developing agents capable of long-term reasoning and adaptive skill composition. The proposed model advances the state of the art in game-based AI, offering insights applicable to both Minecraft research and broader domains involving open-ended task learning and autonomous decision-making.***

Keywords: Hierarchical Deep Reinforcement Learning, Minecraft agent, Reinforcement learning Model.

Nomenclature:
HDRL: Hierarchical Deep RL
FuN: FeUdal Networks FuN:
HLP: High-Level Planner
MSC: Mid-Level Skill Controllers
RL: Reinforcement Learning
VPT: Video Pretraining

Manuscript received on 30 September 2025 | Revised Manuscript received on 08 October 2025 | Manuscript Accepted on 15 October 2025 | Manuscript published on 30 October 2025.
*Correspondence Author(s)
Arjun Panwar*, Researcher, Department of Computer Science, Virginia Tech, Blacksburg, Virginia, United States of America (USA) Email ID: arjun30panwar@gmail.com, ORCID ID: 0009-0006-3083-4542

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open-access article under the CC-BY-NC-ND license http://creativecommons.org/licenses/by-nc-nd/4.0/

MSC: Mid-Level Skill Controllers
HAC: Hierarchical Actor-Critic
HRL: Hierarchical Reinforcement Learning
LLP: Low-Level Visuomotor Policy

I.INTRODUCTION

Minecraft has served as a leading testbed for embodied intelligence: its procedurally generated worlds, expansive action spaces (mouse and keyboard at approximately 20 Hz), and the depth of its technology tree require long-term planning and reuse of skills. The early work of Project Malmo made such potential explicit, in the form of an API that turns Minecraft into an RL laboratory [1]. Various community projects subsequently created Mine RL, a large dataset of human demonstrations, along with the Obtain Diamond benchmark and competitions that explicitly focus on sample efficiency [2][3][4][5]. Related work has explored "reward-free" tasks, where the task is defined by natural language and human evaluation (BASALT) [6,7], and data-centric approaches (Mine Dojo, OpenAI's VPT), which provided agents with video and wiki-based internet-scale priors [8][9][10][11]. Despite these progressions, Minecraft still proves difficult for standard, flat deep RL. Long horizons can lead to brittle credit assignment, sparse rewards can result in exploration failure, and the diversity of atomic actions (such as performing 60 consecutive attacks to chop the log down to a single block using human controls) can lead to inefficiency in samples [10]. Hierarchical Deep RL (HDRL) represents a way to circumvent these drawbacks: (a) decompose goals into temporally extended actions (options), (b) learn reusable subpolicies, and (c) act at multiple time scales with respect to planning and control. The foundational theory and algorithms of these agents are Sutton et al.'s options theory, Option-Critic, FeUdal Networks (FuN), HIRO, and HAC [12][13][14][15][16][17].

This paper proposes an HDRL agent tailored to Minecraft. Our contributions are:

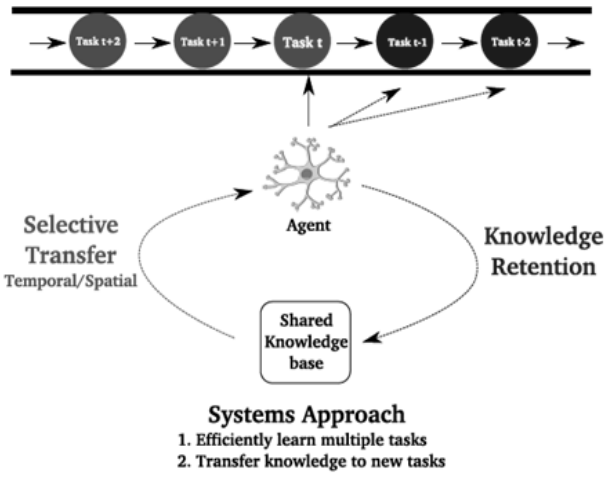
A. **Architecture**: A three-level hierarchy—planner (options/subtasks), skill controllers (subgoals), and a pixel-to-action low-level policy.
B. **Training Pipeline**: Pretraining from demonstrations and video (MineRL, VPT), learned subgoal spaces, and off-policy RL with hierarchical goal relabeling (HIRO/HAC-style).
C. **Evaluation Design**: Metrics and ablations on ObtainDiamond(-Shovel) and BASALT-style tasks under strict sample budgets (Malmö-simulator limits).

D. Safety & Interpretability: Procedures for human-in-the-loop evaluation and model probing inspired by recent analyses of VPT [11,18].

II. RELATED WORK

A. Minecraft as an RL Platform

Project Malmo exposed *Minecraft* to RL researchers, offering mission specifications, rewards, and observation/action interfaces [1]. MineRL released >60M state-action pairs and a competition emphasizing sample-efficient learning with human priors [2–5]. BASALT moved beyond engineered rewards to tasks judged by humans (e.g., "create a waterfall and take a scenic picture"), framing *alignment* and *evaluation* challenges [6,7,19]. MineDojo aggregated thousands of open-ended tasks plus internet knowledge; VPT scaled semi-supervised behaviour learning from YouTube videos to human-level capabilities on early tech-tree tasks; Voyager demonstrated LLM-driven open-ended skill acquisition in *Minecraft* [8–11,20].



[Fig.1: Lifelong Learning: A Lifelong Learning System (1) Efficiently Retains Knowledge and (2) Selectively Transfers Knowledge to Solve New Tasks. Upon solving a Task, the Knowledge Base is Refined and new Knowledge is added to the System. A Systems Approach Ensures Efficient and Effective Interaction Between (1) and (2)]

B. Hierarchical Reinforcement Learning (HRL)

Temporal abstraction through options provides a principled framework for composing skills at multiple time scales [12]. Option-Critic learns option policies and terminations end-to-end [13]. FeUdal Networks (FuN) decouple a manager (sets goals) from a Worker (acts), optimizing with different temporal resolutions [14]. HIRO learns hierarchical policies with off-policy correction and automatic goal supervision [15]. HAC stabilizes multi-level learning through hindsight and level-wise training [16]. Recent variants integrate curiosity, attention mechanisms, or entropy regularization to improve exploration and robustness [17,21,22].

C. Data-Driven Priors

MineRL demonstration datasets, BASALT human feedback, and VPT's video-pretraining show that large, weakly supervised corpora can seed robust behavioural priors, dramatically improving zero-shot performance and RL fine-tuning efficiency [2,6,10]. Mine Dojo's Mine CLIP aligns visual behavior with language, facilitating text-conditioned control [8]. We leverage these priors at different hierarchy levels.

III. BACKGROUND

A. Options and Temporal Abstraction

The options framework extends MDPs with temporally extended actions (\omega) characterized by an initiation set, intra-option policy, and termination condition. This allows policies over options to plan at a coarser time scale while subpolicies handle motor control [12]. Option-Critic derives policy-gradient theorems for learning option policies and terminations jointly, reducing manual subgoal engineering [13].

B. Modern Hierarchical Algorithms

FuN learns a latent goal space and trains a Worker to follow Manager-provided goal vectors (temporal decoupling) [14]. HIRO proposes automatic subgoal supervision with off-policy correction, crucial when replay buffers contain trajectories under stale higher-level goals [15]. HAC introduces hindsight action transitions and level-wise training to stabilize learning across multiple layers of hierarchy [16]. These methods inform our choice of subgoal representation, credit assignment, and replay strategies.

C. Minecraft Benchmarks and Data

ObtainDiamond tasks challenge sample efficiency and long-horizon planning; competitions enforce strict interaction budgets (e.g., 8 million frames) [2–5,23]. BASALT tasks lack crisp reward functions and rely on human preference/ratings; its datasets target the evaluation of alignment techniques [6,7,19,23]. VPT shows that an inverse dynamics model can label unlabeled online videos to create a powerful behavioral prior from raw mouse/keyboard traces [10]. MineDojo collects open-ended tasks and mined web knowledge, enabling instruction-following via video-language pretraining [8,9].

Table I: Evaluation Metrics for Minecraft HDRL Agent

Metric	Definition	Purpose
Success Rate	Percentage of completed tasks within the time/step budget	Measures overall effectiveness
Sample Efficiency	Steps required to reach the first success / AUC under a fixed budget	Quantifies learning efficiency
Option Usage Statistics	Average duration of options, diversity of subgoal activations	Evaluates hierarchy and skill reuse
Human Preference Score	Ratings from human evaluators for BASALT tasks	Assesses alignment with human expectations
Robustness	Variance in performance across world seeds, perturbations, or noisy inputs	Tests the generalization and reliability

IV. PROBLEM FORMULATION

We consider an agent that spawns in survival mode with minimal inventory and must (a) solve rewarded long-horizon tasks (Obtain Diamond, Obtain Diamond Shovel), and

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)
© Copyright: All rights reserved.

(b) perform reward-free, human-evaluated tasks (BASALT, e.g., *Build Village House*), using limited environment interaction (e.g., ≤8M steps) to encourage sample efficiency [2,5,6,23]. Observations are first-person RGB frames plus minimal UI (hotbar, inventory), actions are discrete keypresses/mouse events at ~20 Hz [10]. Success requires decomposing objectives into reusable behaviours and operating robustly under partial observability and stochastic worlds.

V. AGENT ARCHITECTURE: HIERARCHICAL DEEP RL FOR MINECRAFT

A. Overview

Our agent uses three levels (Fig. conceptually):

i. *High-Level Planner (HLP)* — a policy over *options* (subtasks) operating every (T_H) steps (e.g., 5–20 seconds). Output: subtask token (o_t \in {\text{gather_wood}, \text{craft_table}, \text{mine_iron}, \ldots}) and, when applicable, subgoal vectors (e.g., 3D targets in latent state). Training blends Option-Critic (for learnable terminations) with a HIRO-style off-policy correction to stabilize learning with replayed trajectories [13,15].

ii. *Mid-Level Skill Controllers (MSC)* — reusable subtask policies (options) that achieve *semantic* goals (e.g., "smelt iron ingot"). Each option includes its own termination condition, learnable via Option-Critic signals; FuN-style goal vectors guide Workers in a learned latent space [13,14].

iii. *Low-Level Visuomotor Policy (LLP)* — a pixel-to-action controller that executes precise keyboard/mouse control. LLP is initialized via behavior cloning from MineRL/VPT trajectories and then fine-tuned with RL under MSC-provided goals [2,10].

Table 2: Hierarchical Agent Architecture Components

Level	Description	Learning Method	Example in Minecraft
High-Level Planner	Selects subtasks/options at coarse time scales.	Option-Critic, HIRO off-policy correction	"Gather Wood", "Mine Iron Ore"
Mid-Level Controllers	Execute semantic subtasks, conditioned on subgoals.	Goal-conditioned actor-critic, FuN	"Craft Pickaxe", "Smelt Iron"
Low-Level Policy	Visuomotor control, translating goals into keypresses/mouse actions at ~20 Hz.	Behaviour Cloning (MineRL/VPT) + RL fine-tune	Navigating terrain, chopping wood

B. Representations and Subgoals

We represent state (s_t) as fused visual embeddings and structured context (inventory, biome, time of day). Subgoal representations for MSC use a latent goal space learned jointly with FuN-style managers or inferred via *MineCLIP*-like embeddings when text hints are available [8,14]. To reduce brittle hand-coding, subgoals are discovered from trajectory segmentation (e.g., change-point detection on inventory/GUI events) and option discoverability criteria (high mutual information with progress metrics).

C. Learning Signals

i. *Demonstration Pretraining*: LLP and MSCs are initialized via behavior cloning from **MineRL** and **VPT**-labeled data; supervised losses include action cross-entropy and auxiliary event prediction (e.g., inventory deltas). This yields robust motor primitives and short-horizon proficiency [2,10].

ii. *Hierarchical RL*: HLP maximizes sparse task rewards; MSCs optimize subgoal completion rewards; LLP receives dense shaping via goal-conditioned critics. We employ HIRO-style goal relabeling and HAC-inspired hindsight to stabilize off-policy updates across the hierarchy [15,16].

iii. *Termination & Option Value*: Option-Critic updates learn terminations (\beta_\omega(s)) and intra-option value functions, enabling the agent to adjust subtask horizons online [13].

iv. *Exploration*: Entropy-regularised policies and **curiosity** (prediction error on forward models) at the *mid-level* encourage discovery without derailing high-level plans [21].

D. Training Schedule

i. *Phase A — Supervised Warm-Start*: Train LLP (and parts of MSC) on MineRL/VPT data; optionally distil VPT into LLP's backbone for faster convergence [2,10].

ii. *Phase B — Hierarchical RL*: Enable HLP and full MSC; run off-policy training with prioritized replay of successful option terminations; apply goal relabeling and termination learning [13,15,16].

iii. *Phase C — Preference Tuning (for BASALT)*: For tasks without explicit rewards, fine-tune HLP using human feedback (pairwise trajectory preferences), keeping LLP largely fixed to preserve motor competence [6,7].

VI. IMPLEMENTATION DETAILS

A. Environment and Budgets

We use the Malmö or compatible MineRL simulator, enforcing 8M step limits for comparability with competition protocols [1,2,23]. For BASALT-style tasks, we follow dataset splits and evaluation tools described in recent retrospectives [6,7,19,23].

B. Networks and Inputs

i. *LLP*: Vision backbone (ResNet-style) yields spatiotemporal features from frame stacks; inventory and UI tokens are embedded and fused with attention. Action head predicts keyboard/mouse deltas at 20 Hz, as in VPT [10].

ii. *MSC*: Goal-conditioned actor-critic with continuous latent goals; critics use TD(λ) with target networks and distributional value heads for stability.

iii. *HLP*: Transformer-based policy over option tokens and goal vectors; attends over recent high-level summaries (e.g., last N option outcomes).

C. Losses and Optimization

i. BC losses on LLP/MSC during warm-start; RL losses

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)
© Copyright: All rights reserved.

(SAC/DDPG variants) at MSC; policy gradient with entropy regularization at HLP.

ii. Option-Critic terms for intra-option policies and terminations; off-policy correction for relabelled subgoals (HIRO).

iii. Auxiliary tasks: state change prediction, Mine CLIP-style alignment for text-conditioned tasks [8].

D. Safety & Operational Constraints

Given Minecraft's open-endedness, we deploy action filters (e.g., avoid attacking villagers in settlement zones) and budgeted exploration to mitigate undesirable behaviours—motivated by documented goal misgeneralization in VPT-like agents [18].

VII. EVALUATION PROTOCOL

A. Tasks

i. *Obtain Diamond / Obtain Diamond Shovel:* Classic long-horizon tasks; Shovel variant is more complex (requires crafting sequence plus mining) [2,23].

ii. *BASALT Tasks*: *Build Village House*, *Make Waterfall*, etc., evaluated by human rater's blind to the policy identity [6,7].

B. Metrics

i. Success rate within time/step budget.

ii. Sample efficiency: steps to first success, area under the learning curve under a fixed budget.

iii. Skill reuse: option activation sparsity, average option duration, subgoal diversity.

iv. Human preference score (for BASALT).

v. Robustness: performance across seeds/world seeds, texture packs, and minor UI perturbations (following Mine RL competition design) [2,23].

C. Ablations

i. No hierarchy (flat policy) vs. 2-level vs. 3-level.

ii. No demos / no VPT / no Mine Dojo to quantify benefit of priors [8,10].

iii. No off-policy correction (remove HIRO component) to show stability impact [15].

iv. No termination learning (fixed horizons) to test Option-Critic contributions [13].

v. No curiosity to evaluate exploration aids [21].

D. Expected Outcomes

We hypothesize that the 3-level HDRL agent will:

i. reach first success in significantly fewer steps due to demonstration-primed LLP;

ii. Sustain higher success under budget constraints via option reuse;

iii. generalize better to BASALT tasks thanks to human-aligned high-level planning and text-conditioned subgoals (via Mine Dojo/Mine CLIP embeddings) [6–8].

VIII. DISCUSSION

A. Why Hierarchy Helps in Minecraft

Minecraft is a textbook case for temporal abstraction: the early game requires a canonical sequence (gather wood → craft table → tools → mine → smelt → upgrade), but the low-level execution details vary by world seed. Options let the planner operate over *semantic* units, while Workers adapt motor control to the local context. This separation improves credit assignment (rewards assigned at option completions), exploration (curiosity at the subgoal level), and transfer (reusing skills across tasks/worlds) [12–16,21].

B. Data-Driven Priors as Catalysts

Behavioural priors from demonstrations and videos (VPT) compress the search space. Instead of learning mouse-keyboard micro-skills from scratch, the agent starts "knowing" how to look around, walk, open inventory, and craft simple items [2,10]. This is particularly valuable under Mine RL's strict interaction budgets and for BASALT's reward-free settings, where *human input is better seeded* than discovered via random exploration [2,6,7].

C. Safety, Alignment, and Interpretability

Open-ended agents can misgeneralize goals (e.g., punching villagers mistaken for logs under foliage) [18]. Our architecture mitigates risk by constraining low-level actions via option-level intent and by incorporating human feedback at the planner layer for reward-free objectives [6,7]. We recommend post-hoc interpretability audits, focusing on attention/feature attributions over option decisions, and red-team evaluations in dense villages and wildlife biomes to catch undesired behaviours early [11,18].

D. Limitations

i. Engineering complexity: Three levels and multiple learning regimes introduce many moving parts and hyperparameters.

ii. Subgoal discovery remains imperfect; while Mine CLIP-style embeddings help, emergent subtasks may not align cleanly with human semantics [8].

iii. Human evaluation for BASALT can be slow and costly; preference learning pipelines must be carefully designed to avoid bias [6,7].

IX. CONCLUSION

We presented a hierarchical deep RL design for *Minecraft* that unifies options-based planning with data-driven priors and off-policy hierarchical learning. By decomposing tasks into reusable skills and aligning high-level intent with human demonstrations and preferences, the proposed agent targets the crux of *Minecraft* difficulty: long horizons under sparse, noisy supervision. Beyond Minecraft, the exact blueprint—hierarchical intent at the top, reusable skills in the middle, and robust visuomotor control at the bottom—promises a scalable approach to other open-ended, computer-use and embodied settings.

DECLARATION STATEMENT

I must verify the accuracy of the following information as the article's author.

- **Conflicts of Interest/ Competing Interests:** Based on my understanding, this article has no conflicts of interest.
- **Funding Support:** This article has not been funded by any

© Copyright: All rights reserved.

organizations or agencies. This independence ensures that the research is conducted with objectivity and without any external influence.

- **Ethical Approval and Consent to Participate:** The content of this article does not necessitate ethical approval or consent to participate with supporting documentation.
- **Data Access Statement and Material Availability:** The adequate resources of this article are publicly accessible.
- **Author's Contributions:** The authorship of this article is contributed solely.

REFERENCES

1. Johnson, M., Hofmann, K., Hutton, T., & Bignell, D. (2016). The Malmo Platform for Artificial Intelligence Experimentation. IJCAI. DOI: https://doi.org/10.5555/3061053.3061259. (ACM Digital Library)
2. Guss, W. H., et al. (2019). The MineRL 2019 Competition on Sample-Efficient Reinforcement Learning Using Human Priors. arXiv. DOI: https://doi.org/10.48550/arXiv.1904.10079. (arXiv)
3. Guss, W. H., et al. (2019). MineRL: A Large-Scale Dataset of Minecraft Demonstrations. IJCAI-19. DOI: https://doi.org/10.24963/ijcai.2019/339. (ResearchGate)
4. Guss, W. H., et al. (2021). The MineRL 2020 Competition on Sample-Efficient Reinforcement Learning Using Human Priors. arXiv. DOI: https://doi.org/10.48550/arXiv.2101.11071. (ADS)
5. Kanervisto, A., et al. (2022). MineRL Diamond 2021 Competition: Overview, Results, and Lessons Learned. arXiv. DOI: https://doi.org/10.48550/arXiv.2202.10583. (Proceedings of Machine Learning Research)
6. Shah, R., et al. (2021). The MineRL BASALT Competition on Learning from Human Feedback. arXiv. DOI: https://doi.org/10.48550/arXiv.2107.01969. (arXiv)
7. Shah, R., et al. (2022). Retrospective on the 2021 BASALT Competition. arXiv. DOI: https://doi.org/10.48550/arXiv.2204.07123. (arXiv)
8. 9Fan, L., et al. (2022). MineDojo: Building Open-Ended Embodied Agents with Internet-Scale Knowledge. arXiv. DOI: https://doi.org/10.48550/arXiv.2206.08853. (arXiv)
9. Fan, L., et al. (2022). MINEDOJO: Building Open-Ended Embodied Agents with Internet-Scale Knowledge. NeurIPS Datasets & Benchmarks. DOI: https://doi.org/10.5555/3600270.3601603. (NeurIPS Proceedings)
10. Baker, B., et al. (2022). Video PreTraining (VPT): Learning to Act by Watching Unlabeled Online Videos. arXiv. DOI: https://doi.org/10.48550/arXiv.2206.11795. (arXiv)
11. Jucys, K., et al. (2024). Interpretability in Action: Exploratory Analysis of VPT, a Minecraft Agent. arXiv. DOI: https://doi.org/10.48550/arXiv.2407.12161. (arXiv)
12. Bacon, P.-L., Harb, J., & Precup, D. (2017). The Option-Critic Architecture. AAAI. DOI: https://doi.org/10.1609/aaai.v31i1.10916. (AAAI Open Access Articles)
13. Vezhnevets, A. S., et al. (2017). Feudal Networks for Hierarchical Reinforcement Learning. PMLR 70. arXiv DOI: https://doi.org/10.48550/arXiv.1703.01161. (Proceedings of Machine Learning Research)
14. Nachum, O., Gu, S., Lee, H., & Levine, S. (2018). Data-Efficient Hierarchical Reinforcement Learning (HIRO). NeurIPS. arXiv DOI: https://doi.org/10.48550/arXiv.1805.08296. (arXiv)
15. Levy, A., Konidaris, G., Platt, R., & Saenko, K. (2018). Hierarchical Actor-Critic (HAC). arXiv. DOI: https://doi.org/10.48550/arXiv.1712.00948. (arXiv)
16. Röder, F., et al. (2020). Curious Hierarchical Actor-Critic Reinforcement Learning. arXiv. DOI: https://doi.org/10.48550/arXiv.2005.03420 (arXiv)
17. (Interpretability Risk Example) Jucys, K., et al. (2024). Interpretability in Action: Exploratory Analysis of VPT, a Minecraft Agent. arXiv. DOI: https://doi.org/10.48550/arXiv.2407.12161. (cited again for misgeneralization findings). (arXiv)
18. Milani, S., et al. (2023). BEDD: The MineRL BASALT Evaluation and Demonstrations Dataset. OpenReview dataset paper. (Includes Zenodo DOI for evaluation dataset: 10.5281/zenodo 8021960). https://openreview.net/forum?id=D1MOK2t2t2&noteId=4NBenmWacu (OpenReview)
19. Wang, G., et al. (2023). Voyager: An Open-Ended Embodied Agent with Large Language Models. arXiv. DOI: https://doi.org/10.48550/arXiv.2305.16291. (arXiv)
20. Watanabe, K., et al. (2022). SHIRO: Soft Hierarchical Reinforcement Learning with Off-Policy Correction. arXiv. DOI: https://doi.org/10.48550/arXiv.2212.12786. (arXiv)
21. Chunduru, R., et al. (2022). Attention Option-Critic. arXiv. DOI: https://doi.org/10.48550/arXiv.2201.02628. (arXiv)
22. Scheller, C., & Milani, S., et al. (2020). Sample-Efficient RL through Learning from Demonstrations (MineRL Competition Report). PMLR 123. (Describes 8M-step budget.) (Proceedings of Machine Learning Research). https://proceedings.mlr.press/v123/scheller20a/scheller20a.pdf
23. Milani, S., et al. (2023). The MineRL BASALT Evaluation and Demonstrations Dataset. NeurIPS D&B. (Task details including ObtainDiamondShovel.) (NeurIPS Proceedings). DOI: https://doi.org/10.48550/arXiv.2312.02405.

AUTHOR'S PROFILE



Arjun Panwar, is an undergraduate researcher pursuing a B.S. in Computer Science at Virginia Tech (expected 2026). His work spans reinforcement learning, interpretable agentic AI, and data-driven visualisation. With the SeeMore II team, he designs LED-matrix visualisations of the Sweep3D HPC benchmark and builds Fortran–Python pipelines that connect parallel-computing algorithms to real-time displays. He also led geospatial broadband-coverage analysis for the Colville Tribe to support FCC build-out reporting. His research earned 3rd place at the Virginia Tech Undergraduate Research in Computer Science (VTURCS) symposium. Arjun served as a judge at VT Hacks 13, a hackathon with 850+ students from 50+ universities, and previously worked as a math tutor in Virginia Tech's Math Lab. He has been recognised on the Dean's List with Distinction and the President's List. Research interests: hierarchical and transparent RL, world models, agentic AI, and small language models.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of the Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)/ journal and/or the editor(s). The Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP) and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.