SENSEI: Semantic Exploration Guided by Foundation Models to Learn Versatile World Models

Cansu Sancaktar[*,1,2] **Christian Gumbsch**[*,1,3,4] **Andrii Zadaianchuk**[1,5]

Pavel Kolev[1] **Georg Martius**[1,2]

[1] Autonomous Learning, University of Tübingen
[2] Empirical Inference, Max Planck Institute for Intelligent Systems
[3] Neuro-Cognitive Modeling, University of Tübingen
[4] Cognitive and Clinical Neuroscience, TU Dresden
[5] VISLab, University of Amsterdam

Abstract

Exploring useful behavior is a keystone of reinforcement learning (RL). Existing approaches to intrinsic motivation, following general principles such as information gain, mostly uncover low-level interactions. In contrast, children's play suggests that they engage in semantically meaningful high-level behavior by imitating or interacting with their caregivers. Recent work has focused on using foundation models to inject these semantic biases into exploration. However, these methods often rely on unrealistic assumptions, such as environments already embedded in language or access to high-level actions. To bridge this gap, we propose SEmaNtically Sensible ExploratIon (SENSEI), a framework to equip model-based RL agents with intrinsic motivation for semantically meaningful behavior. To do so, we distill an intrinsic reward signal of interestingness from Vision Language Model (VLM) annotations. The agent learns to predict and maximize these intrinsic rewards using a world model learned directly from intrinsic rewards, image observations, and low-level actions. We show that in both robotic and video game-like simulations SENSEI manages to discover a variety of meaningful behaviors. We believe SENSEI provides a general tool for integrating feedback from foundation models into autonomous agents, a crucial research direction as openly available VLMs become more powerful.

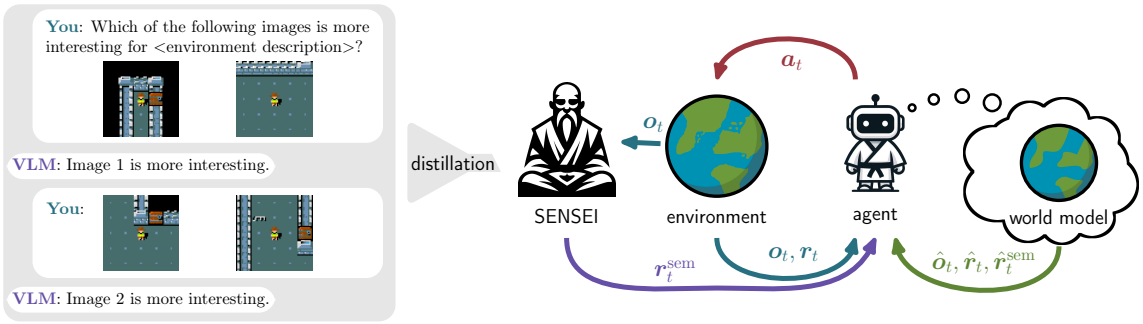


Figure 1: **SENSEI overview**: **(a)** During pre-training we prompt a VLM to compare observations (e.g. images) from an environment with respect to their interestingness. We distill this ranking into a reward function (SENSEI), to guide the exploration of an embodied agent. **(b)** An exploring agent not only receives observations ($\boldsymbol{o}_t$) and rewards (r_t) from interactions with the environment but also a semantic exploration reward (r_t^{sem}) from SENSEI. **(c)** The agent learns a world model from its experience to judge the interestingness ($\hat{r}_t^{\text{sem}}$) of states without querying SENSEI.

*Equal contribution, corresponding authors: cansu.sancaktar@tuebingen.mpg.de, chris@gumbsch.de

1 Introduction

Achieving intrinsically-motivated learning in artificial agents has been a long-standing dream, making it possible to decouple agents' learning from an experimenter manually crafting and setting up tasks. Thus, the goal in intrinsically-motivated reinforcement learning (RL) is for agents to explore their environment efficiently and autonomously, which would constitute a free play phase akin to children's curious play. Various intrinsic reward definitions have been proposed in the literature, such as aiming for state space coverage (Bellemare et al., 2016; Tang et al., 2017; Burda et al., 2019), novelty or retrospective surprise (Pathak et al., 2017; Schmidhuber, 1991), and information gain of a world model (Pathak et al., 2019; Sekar et al., 2020; Sancaktar et al., 2022). However, when an agent starts to learn from scratch, there is one fundamental problem: just because something is novel or unseen does not necessarily mean that it contains useful or generalizable information (Dubey & Griffiths, 2017).

Imagine a robot in front of a desk with objects and drawers. The robot could move its arm through the air to cover its entire manipulable space, or it could hit its end effector on the desk at various speeds, leading to high uncertainty states. In contrast, human common sense would likely focus on interacting with the objects or drawer of the desk since potential task distributions likely revolve around those entities.

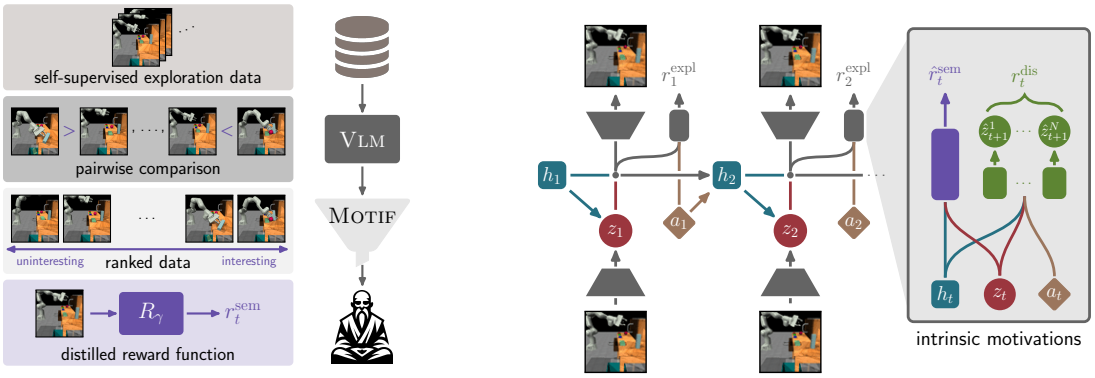
Agents exploring their environment from scratch with intrinsic motivations suffer from a chicken-or-egg problem: *how do you know something is useful before you have tried it and experienced its usefulness to begin with?* This is a bottleneck for the types of behavior that an agent can unlock during free play. We argue that incorporating human priors into exploration could alleviate this roadblock. Similar points have been raised for children's play. During the first years of life, children are surrounded by their caregivers who ideally encourage and reinforce them while they explore their environment. Philosopher and psychologist Karl Groos has stipulated that there is "a strong drive in children to observe the activities of their elders and incorporate those activities into their play" (Gray, 2017; Groos & Baldwin, 1901).

A potential solution in the age of Large Language Models (LLMs), is to utilize language as a cultural-transmitter to inject "human notions of interestingness" (Zhang et al., 2023a) into RL agents' exploration. However, the most prominent works in this domain assume (1) a semantically-grounded environment (Zhang et al., 2023b; Du et al., 2023), (2) in the case of Motif (Klissarov et al., 2023), the availability of an offline dataset with exhaustive state-space coverage and messages labelling the unfolding events or (3) access to high-level actions in the embodied environments considered in Zhang et al. (2023a); Du et al. (2023). These assumptions are still detached from the current reality of embodied agents, e.g. in robotics, which don't come with perfect state or event captioners, with pre-existing offline datasets nor with robust, abstracted away actions. Furthermore, none of these approaches learn an internal model of "interestingness". Thus, they rely on the LLM, or a distilled module, to constantly guide their exploration and fail to transfer this knowledge to novel states when LLM feedback is not available.

In this work we propose SEmaNtically Sensible ExploratIon (Sensei), a framework for Vision Language Models (VLM) guided exploration for model-based RL agents, illustrated in Fig. 1. Sensei starts with a short description of the environment and a dataset of observations (e.g. images) collected through self-supervised exploration. A VLM is prompted to compare the observations pairwise with respect to their interestingness and the resulting ranking is distilled into a reward function. When the agent explores its environment, it receives semantically-grounded exploration rewards from Sensei. The agent learns to predict this exploration signal through its learned world model, corresponding to an internal model of "interestingness", and improves its exploration strategy based on these model-based predictions.

Our main contributions are as follows:

- We propose Sensei, a framework for foundation model-guided exploration with world models.
- We show that Sensei can explore rich, semantically meaningful behaviors with few prerequisites.



(a) reward function distillation

(b) world model

Figure 2: **Intrinsic rewards in SENSEI**: **(a)** Prior to task-free exploration, we prompt GPT-4 to compare images with respect to the interestingness for a certain environment. From the resulting ranking we distill a reward function R_γ using MOTIF. **(b)** Later, an agent learns an RSSM world model from task-free exploration. From each model state, the agent predicts different sources of intrinsic rewards, i.e. epistemic uncertainty-based reward and our distilled semantic reward.

2 Method

We consider the setup of an agent interacting with a Partially Observable Markov Decision Process. At each time t, the agent performs an action $\boldsymbol{a}_t \in \mathcal{A}$ and receives an observation $\boldsymbol{o}_t \in \mathcal{O}$, composed of an image and potentially additional information. We assume that there exist one or more tasks in the environment for which the agent may receive rewards $\boldsymbol{r}_t^{\text{task}} \in \mathbb{R}$ after executing an action. However, during task-free exploration, the agent should select its behavior agnostic to task rewards.

We assume that SENSEI starts with a short description of the environment, a dataset $\mathcal{D}^{\text{init}} \subset \mathcal{O}$ collected from self-supervised exploration (Sekar et al., 2020; Sancaktar et al., 2022), and access to a pretrained VLM. *Prior* to task-free exploration, SENSEI distills a semantic exploration reward function from VLM annotations (Sec. 2.1). *During* exploration, SENSEI learns a world model (Sec. 2.2) and optimizes an exploration policy through model-based RL and intrinsic reward predictions (Sec. 2.3).

2.1 Reward function distillation: MOTIFate your SENSEI

Prior to task-free exploration, SENSEI needs to distill a semantically grounded intrinsic reward function R_γ with learnable parameters γ based on the preferences of a pretrained VLM. While the overall framework of SENSEI is agnostic to the exact distillation method, we chose to use an extension of MOTIF (Klissarov et al., 2023) (illustrated in Fig. 2a).[1]

MOTIF consists of two phases. In the first phase of **dataset annotation**, the pretrained foundation model is used to compare pairs of observations, creating a dataset of preferences. For this, we prompt the VLM with an environment description and provide pairs of observations from $\mathcal{D}^{\text{init}}$, asking the VLM which image it considers to be more interesting. The annotation function is given by the VLM : $\mathcal{O} \times \mathcal{O} \to \mathcal{Y}$, where $\mathcal{O}$ is the space of observations, and $\mathcal{Y} = \{1, 2, \emptyset\}$ is a space of choices for the first, second or none of the observations. In **reward training** phase, a reward function is derived from the VLM preferences using standard techniques from preference-based RL (Wirth et al., 2017). A cross-entropy loss function is minimized on the dataset of preference pairs to learn a semantically grounded reward model $R_\gamma : \mathcal{O} \to \mathbb{R}$. We use the final semantic reward function R_γ whenever the agent interacts with its environment: the agent not only receives an observation $\boldsymbol{o}_t$ and reward $\boldsymbol{r}_t$ after executing an action $\boldsymbol{a}_t$, but also receives a semantically-grounded exploration reward $r_t^{\text{sem}} \leftarrow R_\gamma(\boldsymbol{o}_t)$ (see Fig. 1, center).

[1] Original MOTIF (Klissarov et al., 2023) assumes an environment where there exist captions that describe the events at each time t. Thus, they can use LLMs to annotate the captions instead of using VLMs to annotate observations.

2.2 World model: Let your SENSEI dream

We assume a model-based setting, i.e. the agent learns a world model from its interactions. Following DreamerV3 (Hafner et al., 2023), we implement the world model as a Recurrent State Space Model (RSSM) (Hafner et al., 2019b). The RSSM with learnable parameters ϕ is computed by

$$\text{Posterior:} \quad \boldsymbol{z}_t \sim q_\phi(\boldsymbol{z}_t \mid \boldsymbol{h}_t, \boldsymbol{o}_t) \tag{1}$$

$$\text{Dynamics:} \quad \boldsymbol{h}_{t+1} = f_\phi(\boldsymbol{a}_t, \boldsymbol{h}_t, \boldsymbol{z}_t) \tag{2}$$

$$\text{Prior:} \quad \hat{\boldsymbol{z}}_{t+1} \sim p_\phi(\hat{\boldsymbol{z}}_{t+1} \mid \boldsymbol{h}_{t+1}) \tag{3}$$

In short, the RSSM encodes all interactions through two latent states, a stochastic state $\boldsymbol{z}_t$ and a deterministic memory $\boldsymbol{h}_t$. At each time t, the RSSM samples a new stochastic state $\boldsymbol{z}_t$ from a posterior distribution q_ϕ computed from the current deterministic state $\boldsymbol{h}_t$ and new observation $\boldsymbol{o}_t$ (Eq. 1). The RSSM updates its deterministic memory $\boldsymbol{h}_{t+1}$ based on the action $\boldsymbol{a}_t$ and previous latent states (Eq. 2). Next, the model predicts the next stochastic state $\hat{\boldsymbol{z}}_{t+1}$ (Eq. 3). Once the new observation $\boldsymbol{o}_{t+1}$ is received, the next posterior q_ϕ is computed and the process is repeated.

Besides encoding dynamics within its latent state, the RSSM is also trained to reconstruct external quantities y_t from its latent state via output heads o_ϕ:

$$\text{Output heads:} \quad \hat{y}_t \sim o_\phi(\hat{y}_t \mid \boldsymbol{h}_t, \boldsymbol{z}_t) \quad \text{with} \quad y_t \in \{\boldsymbol{o}_t, c_t, r_t, r_t^{\text{sem}}\} \tag{4}$$

The RSSM of DreamerV3 (Hafner et al., 2023) reconstructs observations $\boldsymbol{o}_t$, episode continuations c_t, and rewards r_t. For SENSEI, we additionally predict the semantic exploration reward r_t^{sem}. The world model is trained end-to-end to jointly optimize the evidence lower bound (Kingma & Welling, 2021).

Thus, our world model learns to predict semantic interestingness $\hat{r}_t^{\text{sem}}$ of states (see Fig. 1, right). We could base exploration exclusively on this signal. However, we expect to face many local optima when optimizing for this signal and we don't want to only explore a fixed set of behaviors, but ensure that the agent goes for interesting and yet novel states. To overcome this limitation, Klissarov et al. (2023) post-process r_t^{sem} and normalize it by episodic event message counts. As we no longer assume ground-truth countable event captions, we combine our new reward signal with epistemic uncertainty, a quantity that was shown to be an effective objective for model-based exploration (Sekar et al., 2020; Pathak et al., 2017; Sancaktar et al., 2022). Following Plan2Explore (Sekar et al., 2020), we train an ensemble of N models with weights $\{\theta^1, \dots, \theta^N\}$ to predict the next stochastic latent states with

$$\text{Ensemble predictor:} \quad \hat{\boldsymbol{z}}_t^n \sim g_{\theta^n}(\hat{\boldsymbol{z}}_t^n \mid \boldsymbol{h}_t, \boldsymbol{z}_t, \boldsymbol{a}_t). \tag{5}$$

We quantify epistemic uncertainty as ensemble disagreement r_t^{dis}, by computing the variance over the ensemble predictions averaged over latent state dimensions J:

$$r_t^{\text{dis}} = \frac{1}{J} \sum_{j=1}^{J} \text{Var}(\hat{z}_{j,t}^n), \tag{6}$$

Thus, the model learns to predict two intrinsic rewards $(\hat{r}_t^{\text{sem}}, r_t^{\text{dis}})$ for a state-action-pair (Fig. 2b).

2.3 Exploration policy: Unleash your SENSEI

We can use a weighted sum of the two intrinsic reward signals, e.g. $r_t^{\text{sem}} + \beta r_t^{\text{dis}}$, as the overall reward r_t^{expl} for optimizing an exploration policy. However, ideally the weighting of the two signals should dynamically depend on the situation. In uninteresting states we want the agent to mostly focus on optimizing interestingness (via r_t^{sem}). However, once the agent has found an interesting state, we would like the agent to branch out and discover new behavior (via r_t^{dis}). We implement this through two trade-off factors $\beta \in \{\beta^{\text{low}}, \beta^{\text{high}}\}$ with $\beta^{\text{high}} > \beta^{\text{low}}$ between which we switch following

$$r_t^{\text{expl}} = \hat{r}_t^{\text{sem}} + \begin{cases} \beta^{\text{high}} r_t^{\text{dis}}, & \text{if} \quad \hat{r}_t^{\text{sem}} \geq Q_k(\hat{r}^{\text{sem}}) \\ \beta^{\text{low}} r_t^{\text{dis}}, & \text{else.} \end{cases} \tag{7}$$

Q_k computes the $k-$th quantile of $\hat{r}^{\text{sem}}$, which we estimate through an exponential moving average. Thus, until a certain level of $\hat{r}^{\text{sem}}$ is reached, the exploration reward mainly aims at maximizing interestingness. After exceeding this threshold, exploration more strongly favors uncertainty-maximizing behavior. Both trade-off factors β^{low} and β^{high} as well as the quantile k are hyperparameters. We learn the exploration policy based on r_t^{expl} using the DreamerV3 algorithm (Hafner et al., 2023).

3 Related work

Intrinsic rewards are applied either to facilitate exploration in tasks where direct rewards are sparse or in a task-agnostic setting where they help collect diverse data. There are many different reward signals that could be useful for efficient exploration of the environment, such as prediction error (Schmidhuber, 1991; Pathak et al., 2017; Kim et al., 2020), novelty and Bayesian surprise (Storck et al., 1995; Blaes et al., 2019; Paolo et al., 2021), learning progress (Schmidhuber, 1991; Colas et al., 2019; Blaes et al., 2019), empowerment (Klyubin et al., 2005; Mohamed & Jimenez Rezende, 2015), metrics for state-space coverage (Bellemare et al., 2016; Tang et al., 2017; Burda et al., 2019) and regularity (Sancaktar et al., 2024). While effective for low-dimensional observations, such objectives could be more challenging to apply in the case of high-dimensional image observations. An approach for exploration in image-based environments is to employ low-dimensional goal spaces (Colas et al., 2019; OpenAI et al., 2021; Nair et al., 2018; Pong et al., 2019; Zadaianchuk et al., 2021; Mendonca et al., 2021). An alternative, more sample efficient, direction, is to learn latent world models (Hafner et al., 2019a; 2023; Gumbsch et al., 2024) from visual observations and use these world models for model-based exploration (Pathak et al., 2019; Sekar et al., 2020). In particular, Plan2Explore (Sekar et al., 2020) uses ensemble disagreement of latent space dynamics predictions as an intrinsic reward. While this is a very general strategy for exploration, this could be limited in more challenging environments where semantically meaningful or goal-directed behavior (Spelke, 1990) is needed for efficient exploration.

Exploration with foundation models: Recent improvements of in-context learning of LLMs open additional ways to explore using human bias of interestingness during exploration (Klissarov et al., 2023; Du et al., 2023; Zhang et al., 2023a) and skill learning (Colas et al., 2020; 2023; Zhang et al., 2023b). MOTIF (Klissarov et al., 2023) leverages LLMs to generate intrinsic rewards by evaluating pairs of event captions to derive rewards, demonstrating its efficacy in the complex game of NetHack (Küttler et al., 2020). This approach has shown that intrinsic rewards can sometimes outperform direct reward maximization strategies. Similarly, ELLM (Du et al., 2023) uses LLMs to guide RL agents towards goals that are meaningful and useful, based on the agent's current state (represented by text), showing improved task coverage in the Crafter environment (Hafner, 2021). Furthermore, OMNI (Zhang et al., 2023a) introduces a novel method to prioritize tasks by modeling human notions of interestingness using LLMs. Thereby, OMNI enhances the open-ended learning process by focusing on tasks that are not only learnable but also generally interesting. LAMP (Adeniji et al., 2023) proposes to use VLMs for reward modulation in an RL setup. First, a set of potential tasks are generated with an LLM and then LAMP uses VLMs to generate rewards for these tasks to learn a language-conditioned policy in the pretraining phase. This policy is later finetuned with actual task rewards.

Reward-shaping through VLMs: Most works that rely on VLMs as reward sources try to solve the reward specification problem in RL. In these works, often a task is assumed to be described as a language caption (Cui et al., 2022; Rocamonde et al., 2023; Baumli et al., 2023; Adeniji et al., 2023), as a goal image (either in-distribution or out-of-distribution) (Cui et al., 2022), or as a demonstration video of the task (Sontakke et al., 2023). In particular, RL-VLM-F (Wang et al., 2024) uses a very similar setup to ours to generate reward functions. Images of initial rollouts are compared pairwise using a VLM to distill a reward function via MOTIF (Klissarov et al., 2023). However, unlike in our work, the VLM is explicitly prompted with the task, whereas we attempt to distill an environment-specific but general exploration reward. Furthermore, SENSEI assumes a model-based setup to learn a world model instead of optimizing a policy based on the distilled reward function.

4 Results

Our experiments set out to empirically evaluate the following questions:

1. Does the distilled reward function R_γ from VLM annotations encourage interesting behavior?
2. Can SENSEI discover semantically meaningful behavior during task-free exploration?
3. Is the explored world model suitable for later learning to efficiently solve downstream tasks?

We answer these questions by (1) illustrating the semantic rewards, (2) quantifying the behavior discovered by SENSEI during task-free exploration and (3) employing the explored world models to later train task-based policies. We use two very different types of environments:

Robodesk (Kannan et al., 2021) is a multi-task RL benchmark in which a simulated robotic arm can interact with various objects on a desk, including buttons, two types of blocks, a ball, a sliding cabinet, a drawer, and a bin. For different objects, there exist different tasks, e.g. `open_drawer` or `push_flat_block_in_bin`, with individual sparse rewards. Robodesk uses pixel-based observations and continuous actions controlling the end-effector (more details in Suppl. C.1).

MiniHack (Samvelyan et al., 2021) is a sandbox to design RL tasks based on NetHack (Küttler et al., 2020). In MiniHack, an agent needs to navigate dungeons by interacting with its environment in meaningful ways, e.g. apply a key to open a door. We tested two tasks: fetching a key in a huge room to unlock a smaller room with an exit (`KeyRoom-S15`) or fetching a key to open a hidden chest in a maze of rooms (`KeyChest`). MiniHack uses discrete actions. As observations we use pixel-based, ego-centric views around the agent and a binary flag indicating key pick-ups (details in Suppl. C.2).

In both environments, we compare SENSEI to Plan2Explore (Sekar et al., 2020), the current state-of-the art in model-based exploration with pixel-based observations. In Robodesk, we compare different versions of SENSEI. In one ablation, we replace the VLM (GPT-4 Turbo) with a hand-crafted oracle (see Suppl. C.4). Furthermore, we compare two initial datasets $\mathcal{D}^{\text{init}}$ of self-supervised exploration collected either by CEE-US (Sancaktar et al., 2022) or by Plan2Explore. CEE-US uses vector-based position of entities for information-gain-based exploration, in comparison to Plan2Explore, which works on the pixel-level. Due to the priviliged inputs, $\mathcal{D}^{\text{init}}_{\text{CEE-US}}$ contains more complex interactions. We compare 1M steps of exploration with the four versions of SENSEI and Plan2Explore. In Minihack, we analyze exploration of 500k steps and only use GPT-4 as annotator using Plan2Explore data.

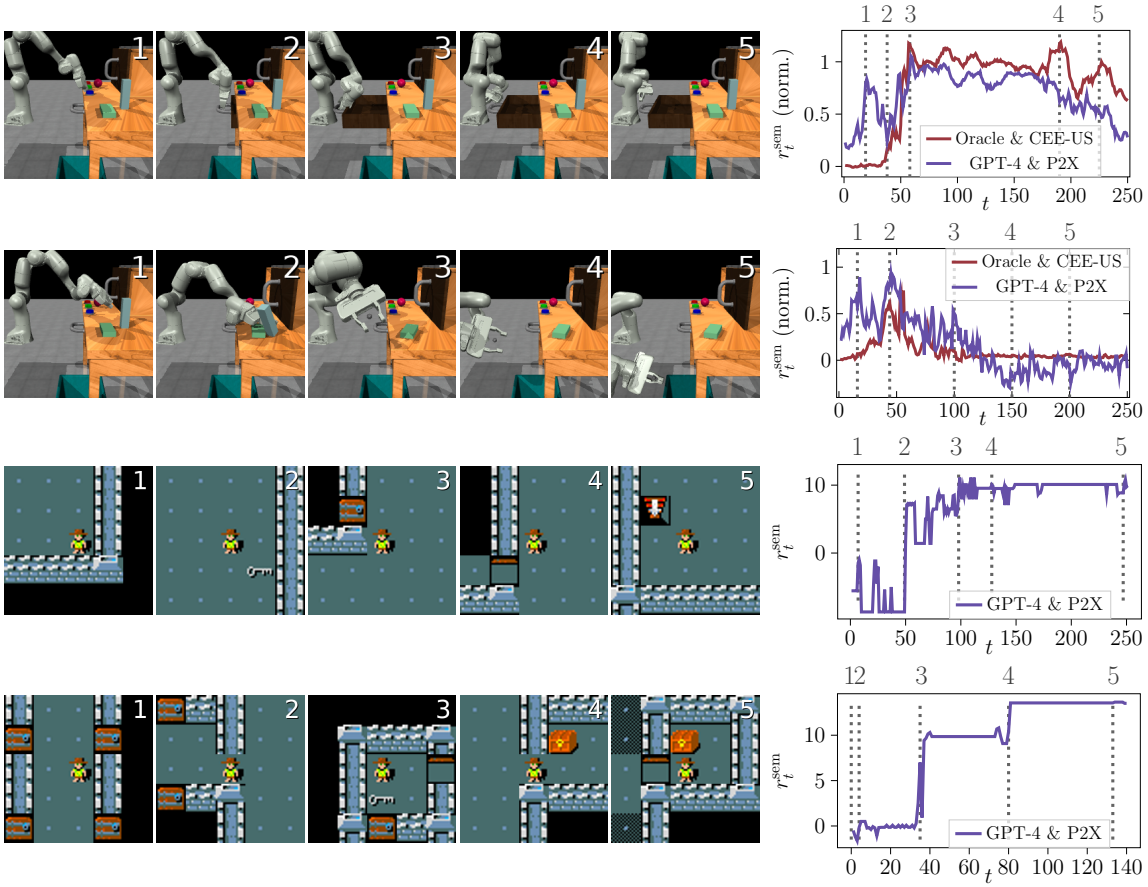
4.1 Reward function of SENSEI

We illustrate how the distilled MOTIF reward function R_γ assigns semantic exploration rewards r^{sem}_t for exemplary sequences from Robodesk and MiniHack. We do this for oracle annotations and GPT-4 annotations in Robodesk and for GPT-4 annotations in MiniHack tasks, as shown in Fig. 3. For Robodesk, we see that as the robot is interacting with objects, r^{sem}_t also increases for the examples of opening the drawer and interacting with the blocks. In the Minihack environments, we clearly see jumps in reward r^{sem}_t for significant events. Frames 2 & 3 in `KeyRoom-S15` and `KeyChest` respectively, are right before the key is picked up. Later, r^{sem}_t increases further once the agent is at the door or chest with a key (Frame 3 in `KeyRoom-S15` and Frames 4&5 in `KeyChest`).

4.2 Task-free exploration

Next, we analyze the interactions uncovered by SENSEI during task-free exploration. We compare 1M steps of exploration in Plan2Explore with the four versions of SENSEI: GPT-4 and oracle as annotators on Plan2Explore and CEE-US exploration datasets.

Fig. 4 plots the mean object interactions during exploration for the different methods. On average, all versions of SENSEI interact more with the objects than Plan2Explore. As a result, SENSEI also solves more tasks during exploration than Plan2Explore (shown in Suppl. D.1). Qualitatively, we observe



(a) screenshots (b) semantic exploration reward

Figure 3: **Semantic exploration rewards for example trajectories**: From top to bottom we show example trajectories for two Robodesk episodes, Minihack `KeyRoom-S15` and `KeyChest`. We showcase rewards from MOTIF distilled from GPT-4 annotations using Plan2Explore (P2X) data (purple). For Robodesk we also show MOTIF distilled from the oracle annotator using CEE-US exploration data (red). The reward trajectories peak at the "interesting" moments of exploration, such as opening a drawer in Robodesk or picking up the key in MiniHack.

Plan2Explore performing mostly arm stretches.[2] Thus, our semantic exploration reward seems to lead to more object interactions than pure epistemic uncertainty-based exploration. SENSEI with oracle shows the most object interactions. The initial exploration dataset $\mathcal{D}^{\text{init}}$ influences with which objects SENSEI interacts. Oracle SENSEI distilled from Plan2Explore data focuses mainly on the sliding cabinet and the upright block, reinforcing the existing trends of Plan2Explore. For CEE-US data, oracle SENSEI interacts more with the other objects and the drawer. If a VLM annotates images instead of the oracle, SENSEI shows similar behavioral trends, but overall less object interactions. We hypothesize that the VLM provides a much noisier signal of interestingness, making it harder to optimize for.

Next, we compare task-free exploration (500k steps) of SENSEI (GPT-4, Plan2Explore data $\mathcal{D}^{\text{init}}$) to Plan2Explore in two tasks of MiniHack. For task-relevant events, the mean number of interactions are plotted in Fig. 5. SENSEI focuses more on semantically interesting interactions, e.g. picking up a key, opening a locked door, or finding the chest with a key, than Plan2Explore. As a result, SENSEI completes both tasks more frequently than Plan2Explore during task-free exploration, as evident by

[2]Interestingly, this can still lead to solving tasks during exploration. For example, stretching the arm against the sliding cabinet can close it, and by stretching the arm towards the upright block, it may be pushed off the table.

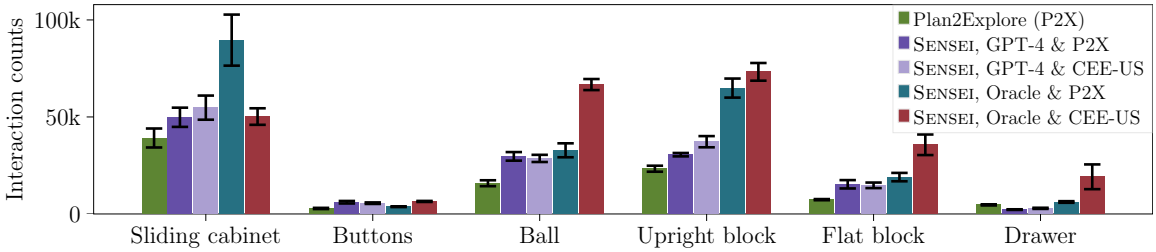


Figure 4: **Interactions in Robodesk**: We plot the mean over the number of interactions with an object during exploration for different versions of SENSEI (Oracle vs. VLM, CEE-US vs. Plan2Explore data $\mathcal{D}^{\text{init}}$) and Plan2Explore. Error bars show the standard deviation (3 seeds).

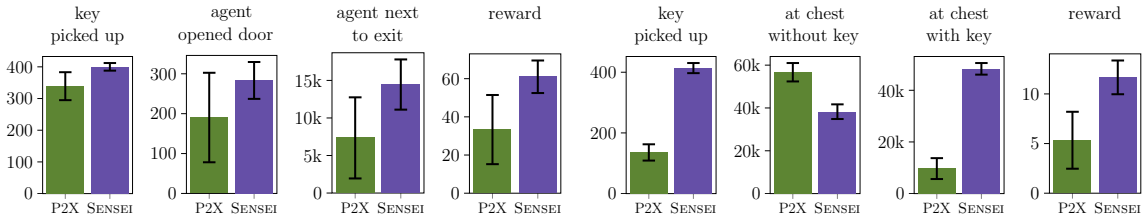
the higher number of collected rewards. We believe that this indicates that SENSEI is well-suited for initial task-free exploration in these environments to discover the sparse rewards.

4.3 Downstream task solving

We hypothesize that world models learned through richer exploration would enable model-based RL agents to learn to solve new downstream tasks in a more sample-efficient manner and with higher success rates. We investigate this in MiniHack by running DreamerV3 (Hafner et al., 2023), where we try to solve the extrinsic tasks using the learned world models from our initial 500K steps of exploration with SENSEI or Plan2Explore (see Sec. 4.2). In this setup, we train a new task-based policy from the imagined rollouts of the world models. We compare the world models learned from Plan2Explore to the world models of SENSEI. Figure 6 shows the performance of the task-based policies over training. Only agents with world models explored through SENSEI can reliably learn to solve the task within 1M steps. As SENSEI allocates more resources to exploring the relevant dynamics in the environment, e.g. opening the chest more frequently instead of just being near the chest, a more versatile world model is learned that aids policy optimization.

5 Discussion

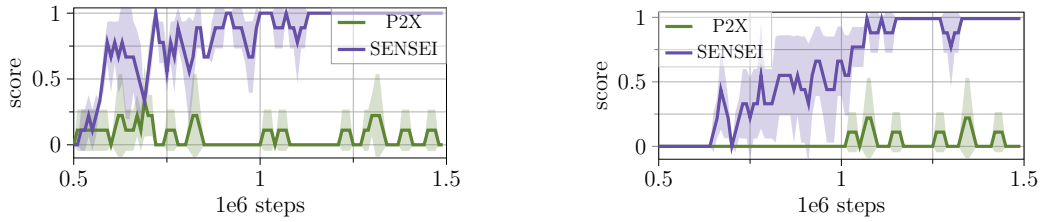
We have introduced SENSEI, a framework for guiding the intrinsically motivated exploration of model-based agents through foundation models without assuming access to expert data, high-level actions, or perfect environment captions. SENSEI bootstraps its model of interestingsness from previously generated play data with e.g. information gain. On this dataset, SENSEI prompts a VLM to compare images with respect to their interestingness and distills a reward function for semantically grounded exploration. SENSEI learns an exploration policy via model-based RL using two sources of intrinsic rewards: (1) trying to reach states with high semantic interestingness and (2) branching out from



(a) interactions and rewards in `KeyRoom-S15`

(b) interactions and rewards in `KeyChest`

Figure 5: **Interactions in MiniHack**: We plot the mean number of interactions with task-relevant objects and the number of rewards collected by SENSEI and Plan2Explore (P2X) for `KeyRoom-S15` (**a**) and `KeyChest` (**b**). Error bars show the standard deviation (3 seeds).



(a) extrinsic phase in `KeyRoom-S15`

(b) extrinsic phase in `KeyChest`

Figure 6: **Downstream task performance in MiniHack**: We plot the mean of the episode score obtained during evaluation for the MiniHack tasks `KeyRoom-S15` (**a**) and `KeyChest` (**b**) with world models learned from SENSEI vs. Plan2Explore (P2X) exploration. Shaded areas depict the standard deviation (3 seeds) and we apply smoothing over the score trajectories with window size 3.

these states to maximize epistemic uncertainty. We show that in a simulated robotic environment, this strategy leads to more object manipulations than pure information gain-oriented exploration. Similarly, we demonstrate that SENSEI discovers more semantically meaningful interactions, such as applying a key to open a chest, when exploring the video game-like environments of MiniHack. In both environments, SENSEI accumulates more rewards than the state-of-the-art exploration method Plan2Explore (Sekar et al., 2020), by "accidentally" solving the tasks already during exploration.

Future work We hypothesize that SENSEI learns a versatile world model from exploration, as it is trained on more interaction-rich and interesting data. In line with past works from model-based planning, we expect this to enable the agent to quickly learn to solve downstream tasks (Sancaktar et al., 2022). We showcase this in the MiniHack tasks `KeyRoom-S15` and `KeyChest`. In future work, we plan to test whether we observe performance gains in a multi-task environment such as Robodesk. Furthermore, by comparing two sources for the initial play data $\mathcal{D}^{\text{init}}$, we observe that SENSEI reinforces the trends existing in the initial exploration round, while still seeing improvements across different types of interactions. To further reinforce more complex types of interactions, our reward function could be refined in a new round of VLM annotations from a SENSEI run. As a result, SENSEI could potentially unlock increasingly complex sequences of behavior with each generation.

Limitations SENSEI benefits from fully-observable observations, e.g. images that capture all relevant aspects of the environment. The VLM annotations, and as a result the distilled reward function, degrade when dealing with occlusions and the lack of depth information from a single image. For example, when the drawer in Robodesk is fully closed and not clearly visible, it is not interesting to just be at the drawer handle (Fig. 3, top, screenshot 2). We believe this can be remedied in future work by adding a second camera angle or using videos for VLM annotations to better convey temporal or partially observable information.

Acknowledgements

The authors thank Sebastian Blaes and Onno Eberhard for helpful discussions. The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Cansu Sancaktar and Christian Gumbsch. Georg Martius is a member of the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645. We acknowledge the financial support from the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039B). This work was supported by the Volkswagen Stiftung (No 98 571).

References

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

2023.

Kate Baumli, Satinder Baveja, Feryal Behbahani, Harris Chan, Gheorghe Comanici, Sebastian Flennerhag, Maxime Gazeau, Kristian Holsheimer, Dan Horgan, Michael Laskin, et al. Vision-language models as a source of rewards. *arXiv preprint arXiv:2312.09187*, 2023.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Sebastian Blaes, Marin Vlastelica, Jia-Jie Zhu, and Georg Martius. Control What You Can: Intrinsically motivated task-planning agent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/b6f97e6f0fd175613910d613d574d0cb-Abstract.html.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=H1lJJnR5Ym.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo Perez-Vicente, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *Advances in Neural Information Processing Systems*, 36, 2024.

Cédric Colas, Pierre Fournier, Mohamed Chetouani, Olivier Sigaud, and Pierre-Yves Oudeyer. CURIOUS: Intrinsically motivated modular multi-goal reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2019. URL https://proceedings.mlr.press/v97/colas19a.html.

Cédric Colas, Tristan Karch, Nicolas Lair, Jean-Michel Dussoux, Clément Moulin-Frier, Peter Dominey, and Pierre-Yves Oudeyer. Language as a cognitive tool to imagine goals in curiosity driven exploration. *Advances in Neural Information Processing Systems*, 33:3761–3774, 2020.

Cédric Colas, Laetitia Teodorescu, Pierre-Yves Oudeyer, Xingdi Yuan, and Marc-Alexandre Côté. Augmenting autotelic agents with large language models. In *Conference on Lifelong Learning Agents*, pp. 205–226. PMLR, 2023.

Yuchen Cui, Scott Niekum, Abhinav Gupta, Vikash Kumar, and Aravind Rajeswaran. Can foundation models perform zero-shot task specification for robot manipulation? In *Learning for Dynamics and Control Conference*, pp. 893–905. PMLR, 2022.

Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models. In *International Conference on Machine Learning*, pp. 8657–8677. PMLR, 2023.

Rachit Dubey and Thomas L. Griffiths. A rational analysis of curiosity. In *Proceedings of the 39th Annual Conference of the Cognitive Science Society*, pp. 307–312, 2017. URL https://cogsci.mindmodeling.org/2017/papers/0068/index.html.

Peter Gray. What exactly is play, and why is it such a powerful vehicle for learning? *Topics in Language Disorders*, 37(3):217–228, 2017.

K. Groos and E.L. Baldwin. *The Play of Man*. Appleton, 1901. URL https://books.google.de/books?id=sR8zAAAAMAAJ.

Christian Gumbsch, Noor Sajid, Georg Martius, and Martin V. Butz. Learning hierarchical world models with adaptive temporal abstractions from discrete latent dynamics. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=TjCDNssXKU.

Danijar Hafner. Benchmarking the spectrum of agent capabilities. *arXiv preprint arXiv:2109.06780*, 2021.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2019a.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, 2019b.

Danijar Hafner, Kuang-Huei Lee, Ian Fischer, and Pieter Abbeel. Deep hierarchical planning from pixels. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104v1*, 2023.

Harini Kannan, Danijar Hafner, Chelsea Finn, and Dumitru Erhan. Robodesk: A multi-task reinforcement learning benchmark. https://github.com/google-research/robodesk, 2021.

Kuno Kim, Megumi Sano, Julian De Freitas, Nick Haber, and Daniel Yamins. Active world model learning with progress curiosity. In *International Conference on Machine Learning (ICML)*, 2020. URL https://arxiv.org/abs/2007.07853.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2021.

Martin Klissarov, Pierluca D'Oro, Shagun Sodhani, Roberta Raileanu, Pierre-Luc Bacon, Pascal Vincent, Amy Zhang, and Mikael Henaff. Motif: Intrinsic motivation from artificial intelligence feedback. *arXiv preprint arXiv:2310.00166*, 9 2023.

A.S. Klyubin, D. Polani, and C.L. Nehaniv. Empowerment: a universal agent-centric measure of control. In *IEEE Congress on Evolutionary Computation*, volume 1, pp. 128–135 Vol.1, 2005. URL https://ieeexplore.ieee.org/document/1554676.

Heinrich Küttler, Nantas Nardelli, Alexander Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. The nethack learning environment. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7671–7684, 2020.

Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and achieving goals via world models. *Advances in Neural Information Processing Systems*, 34: 24379–24391, 2021.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015. URL https://arxiv.org/abs/1509.08731.

Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems*, 31, 2018.

OpenAI, Matthias Plappert, Raul Sampedro, Tao Xu, Ilge Akkaya, Vineet Kosaraju, Peter Welinder, Ruben D'Sa, Arthur Petron, Henrique Ponde de Oliveira Pinto, Alex Paino, Hyeonwoo Noh, Lilian Weng, Qiming Yuan, Casey Chu, and Wojciech Zaremba. Asymmetric self-play for automatic goal discovery in robotic manipulation. *arXiv:2101.04882*, 2021. URL https://arxiv.org/abs/2101.04882.

Giuseppe Paolo, Alexandre Coninx, Stephane Doncieux, and Alban Laflaquière. Sparse reward exploration via novelty search and emitters. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 154–162, 2021. URL https://doi.org/10.1145/3449639.3459314.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, 2017. URL https://proceedings.mlr.press/v70/pathak17a.html.

Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *International Conference on Machine Learning (ICML)*, 2019. URL https://proceedings.mlr.press/v97/pathak19a.html.

Vitchyr H Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

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

Mikayel Samvelyan, Robert Kirk, Vitaly Kurin, Jack Parker-Holder, Minqi Jiang, Eric Hambro, Fabio Petroni, Heinrich Kuttler, Edward Grefenstette, and Tim Rocktäschel. Minihack the planet: A sandbox for open-ended reinforcement learning research. In *Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Cansu Sancaktar, Sebastian Blaes, and Georg Martius. Curious exploration via structured world models yields zero-shot object manipulation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Cansu Sancaktar, Justus Piater, and Georg Martius. Regularity as intrinsic reward for free play. *Advances in Neural Information Processing Systems*, 36, 2024.

Jürgen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proceedings of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, 1991. URL https://dl.acm.org/doi/10.5555/116517.116542.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning (ICML)*, 2020. URL https://proceedings.mlr.press/v119/sekar20a.html.

Sumedh A Sontakke, Jesse Zhang, Sébastien MR Arnold, Karl Pertsch, Erdem Bıyık, Dorsa Sadigh, Chelsea Finn, and Laurent Itti. Roboclip: one demonstration is enough to learn robot policies. *arXiv preprint arXiv:2310.07899*, 2023.

Elizabeth S Spelke. Principles of object perception. *Cognitive science*, 14(1):29–56, 1990.

J. Storck, S. Hochreiter, and J. Schmidhuber. Reinforcement driven information acquisition in non-deterministic environments. In *Proceedings of the International Conference on Artificial Neural Networks*, pp. 159–164, Paris, 1995. EC2 & Cie. URL https://people.idsia.ch/~juergen/icann95new.pdf.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. *arXiv preprint arXiv:2402.03681*, 2024.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 2017.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on Robot Learning*, pp. 2226–2240. PMLR, 2023.

Andrii Zadaianchuk, Maximilian Seitzer, and Georg Martius. Self-supervised visual reinforcement learning with object-centric representations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=xppLmXCbOw1.

Jenny Zhang, Joel Lehman, Kenneth Stanley, and Jeff Clune. Omni: Open-endedness via models of human notions of interestingness. *arXiv preprint arXiv:2306.01711*, 2023a.

Jesse Zhang, Jiahui Zhang, Karl Pertsch, Ziyi Liu, Xiang Ren, Minsuk Chang, Shao-Hua Sun, and Joseph J Lim. Bootstrap your own skills: Learning to solve new tasks with large language model guidance. In *7th Annual Conference on Robot Learning*, 2023b. URL https://openreview.net/forum?id=a0mFRgadGO.

Supplementary Material for: SENSEI: Semantic Exploration Guided by Foundation Models to Learn Versatile World Models

A SENSEI: Implementation Details

A.1 World model

RSSM We base our RSSM implementation on DreamerV3 (Hafner et al., 2023). For MiniHack we use the small model size setting with roughly 18M parameters ($\boldsymbol{h}_t$ dimensions: 512, CNN multiplier: 32, dense hidden units: 512, MLP layers: 2). For the more complicated Robodesk environment, we use the medium model size with around 37M parameters ($\boldsymbol{h}_t$ dimensions: 1024, CNN multiplier: 48, dense hidden units: 640, MLP layers: 3). By default, when the input observation $\boldsymbol{o}_t$ is only an image, it is en- and decoded through CNNs. For MiniHack, we have an additional inventory flag that is processed by a separate MLP, as is customary for the Dreamer line of work when dealing with multimodal inputs (Wu et al., 2023). The MLP decoder outputs a Bernoulli distribution from which we sample the decoded inventory flag.

Reward predictors To handle rewards of widely varying magnitudes, DreamerV3 uses twohot codes predicted in symlog space when predicting rewards (Hafner et al., 2023). We use the same setup for all reward prediction heads, i.e. for extrinsic rewards r_t^i for task i or the semantic exploration reward r_t^{sem}. During task-free exploration, the gradients from reward predictions are stopped to not further affect world model training. We do this to keep the world model somewhat task-agnostic to later reuse it for multiple tasks. Similarly, to avoid overfitting to the exploration regime, we also stop the gradients from the semantic reward prediction heads.

Plan2Explore Both our Plan2Explore baseline as well as our ensemble predictors (Eq. 5) are based on the re-implementation on top of DreamerV3. The most notable difference is that in original Plan2Explore the ensemble is trained to predict image encodings (Sekar et al., 2020), whereas the new version is trained to predict stochastic states $\boldsymbol{z}_t$. Recent re-implementations (Hafner, 2021; Hafner et al., 2022; Gumbsch et al., 2024) also used Plan2Explore with ensemble disagreement over $\boldsymbol{z}_t$ as a baseline and verified a strong exploration performance.

Quantile estimation We update our estimate of the quantile $Q_k(\hat{r}^{\text{sem}})$ whenever we train the exploration policy. For this, we compute the k-th quantile of $\hat{r}_t^{\text{sem}}$ in each training batch (16×16). We keep an exponential moving average over these estimates with a smoothing factor of $\alpha = 0.99$.

Reward weighting In practice, we compute exploration rewards (Eq. 8) using two reward factors for each loss term

$$r_t^{\text{expl}} = \begin{cases} \alpha^{\text{high}} \hat{r}_t^{\text{sem}} + \beta^{\text{high}} r_t^{\text{dis}}, & \text{if} \quad \hat{r}_t^{\text{sem}} \geq Q_k(\hat{r}^{\text{sem}}) \\ \alpha^{\text{low}} \hat{r}_t^{\text{sem}} + \beta^{\text{low}} r_t^{\text{dis}}, & \text{else,} \end{cases} \tag{8}$$

i.e. α to scale $\hat{r}_t^{\text{sem}}$ and β to scale r_t^{dis}. When training the value function with DreamerV3, the scale of the reward sources are normalized. To compute this normalization for the exploration policy we use α^{low} and β^{low} of the low percentile region of interestingness ($\leq Q_k$). Thus, if $\alpha^{\text{high}} + \beta^{\text{high}} > \alpha^{\text{low}} + \beta^{\text{low}}$ then rewards of the high percentile region ($\geq Q_k$) can exceed the normalization bound. We believe this can act as an additional exploration bonus, since the exploration policy can gain more overall rewards in regions with high semantic interestingness.

A.2 Semantic Reward Distillation: Motif

For the semantic reward function $R_\gamma : \mathcal{O} \to \mathbb{R}$, we use a 2D-convolutional neural network to encode the images. We use 3 convolutional layers, where we progressively increase the number of channels

to `num_channels_max` = 64. The output then gets downsampled via max pooling before going into a two-layer MLP with hidden dimensions 256 & 512 and outputting the scalar reward value. Additionally, in MiniHack we include inventory information via a separate multi-layer perceptron (MLP) head, consisting of 2 layers with 512 hidden units. The extracted features are concatenated with the image features and get further processed by the output MLP. The training hyperparameters for all R_γ can be found in Suppl. B.

B Hyperparameters

We provide the hyperparameters used for the world model, exploration policy, MOTIF annotations & reward model training as well as the environment-specific settings.

Name		**Value**	
	Robodesk	KeyRoom	KeyChest
World Model			
RSSM size	M	S	S
Ensemble size N	8	8	8
Exploration policy			
Quantile	0.85 - 0.75 -0.75 - 0.80	0.90	0.90
α^{high}	0.1 - 0.1 - 0.05 - 0.01	0.3	0.25
β^{high}	1 - 1 - 1 - 1	1	1
α^{low}	1 - 1 - 1 - 1	1	1
β^{low}	0 - 0 - 0 - 0	0.1	0.05
Annotations for MOTIF			
VLM	GPT-4 turbo	GPT-4 omni	GPT-4 omni
Temperature	0.2	0.2	0.2
Dataset size	200K	100K	100K
Image res.	224×224	80×80	80×80
MOTIF Training			
Batch size	64 - 64 - 32 - 32	32	32
Learning rate	10^{-5} - 10^{-5}- 3×10^{-5} - 3×10^{-5}	10^{-4}	10^{-4}
Weight decay	0 - 0 - 0 - 0	10^{-5}	10^{-4}
Environment			
Action repeat	2	1	1
Episode length	250	600	800
Steps of exploration	1M	500K	500k

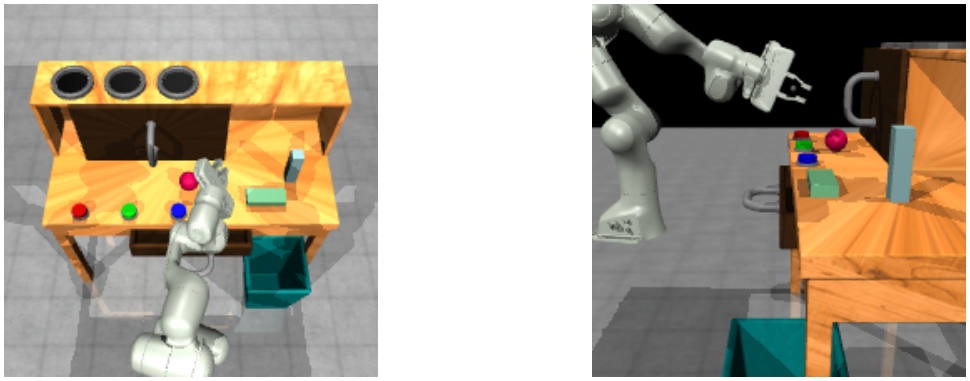
For the exploration policy in Robodesk we use different values for the four different variants tested. The values listed here stand for, from left to right: GPT-4 with Plan2Explore data, GPT-4 with CEE-US data, Oracle with Plan2Explore data, and Oracle with CEE-US data. The MOTIF training hyperparameters are also listed in the same order.

Note that for the world model we use 64×64 images for all environments. However, for the GPT annotations we use higher resolution images, as shown in the table. Inside the environment `step` function, the rendering is performed at these higher resolutions, and this image is input to the semantic reward function R_γ. The image is then scaled down to 64×64 as part of the observation that the RSSM is trained on.

C Environment Details

C.1 Robodesk

Robodesk (Kannan et al., 2021) is a multi-task RL benchmark in which a robot can interact with various objects on a desk. We use an episode length of 250 time steps.



(a) Default observations (b) Our observations

Figure 7: **Robodesk environment.** We modify the default top-down camera view **(a)** to a side view with less occlusion **(b)**.

Observations Robodesk uses only an image observation, depicting the current scene, which we scale down (64×64 pixels). However, we found that the default top-down view often had occlusions and was hard to interpret from a single image (Fig. 7a). Thus, we used a different camera angle showing the robot from one side (Fig. 7b). With this view objects and the drawer were rarely occluded; however, lights that turn on from button presses were not visible anymore.

Actions The continuous 5-dimensional actions control the movement of the end effector. We use an action repeat of 2 to speed up the simulation. Thus, 1M steps of exploration correspond to 2M actions in the environment.

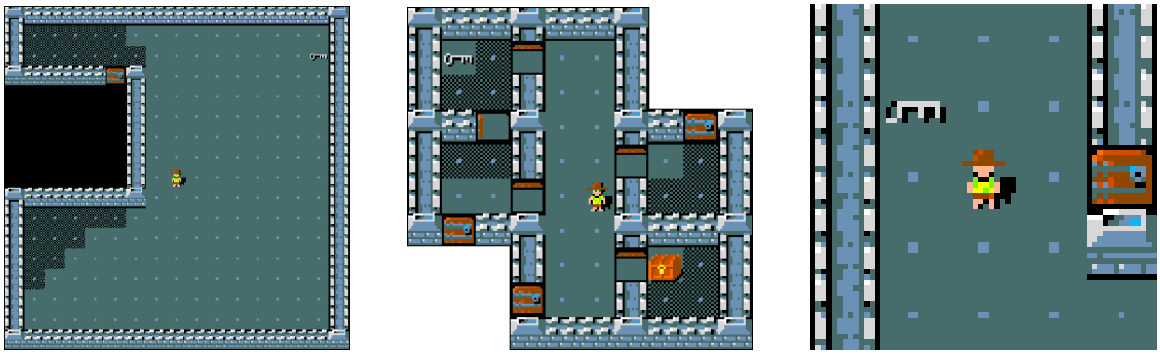
Interaction metrics We track how often the robot interacted with different objects to quantify the behavior during exploration by tracking the velocity of joints and object positions. For buttons, sliding cabinet, or drawer, we check if the joint position changes more than a fixed value (0.02). For all other objects, we check if any of their x-y-z velocities exceed a threshold (0.02).

Tasks We use the sparse reward versions of all the tasks available in the environment. For some tasks, we add easier versions. All tasks describe interactions with one or multiple objects:

- **Buttons**: Pushing the red (`push_red`), blue (`push_blue`), or green (`push_green`) button.
- **Sliding cabinet**: Opening the sliding cabinet fully (`open_slide`) or partially (`open_slide_easy`). The latter is a task variant added by us.
- **Drawer**: Opening the drawer fully (`open_drawer`), considerably (`open_drawer_medium`) or opening it slightly (`open_drawer_easy`). The latter two tasks were added by us.
- **Upright Block**: Lifting the upright block (`lift_upright_block`), pushing it off the table (`upright_block_off_table`) or putting it into the shelf (`upright_block_in shelf`).
- **Flat Block**: Lifting the flat block (`lift_flat_block`), pushing it off the table (`flat_block_off_table`), into the bin (`flat_block_in_bin`), or into the shelf (`flat_block_in_shelf`).
- **Both blocks**: Stacking both blocks (`stack`).
- **Ball**: Lifting the ball (`lift_ball`), dropping it into the bin (`ball_in_bin`) or putting it into the shelf (`ball_in_shelf`).

C.2 MiniHack

Observations In MiniHack multiple observation and action spaces are possible. We use ego-centric, pixel-based observations centered on the agent (± 2 grids, example in Fig. 8c). In addition to that,



(a) `KeyRoom-S15` (b) `KeyChest` (c) egocentric view

Figure 8: **MiniHack** : We consider two tasks `KeyRoom-S15` **(a)** and `KeyChest` **(b)**. The agent receives an egocentric view of the environment as its observation **(c)**.

we provide the agent's inventory. By default, in MiniHack the inventory is given as an array of strings (UTF8 encoded), and different player characters have different starting equipment based on the character classes of NetHack. We simplify this by providing only a binary flag that indicates if the agent has picked up a new item. This is sufficient for the problems we consider, in which maximally one new item can be collected and starting equipment cannot be used.

Environments Here we detail the environments we tackle:

In the benchmark `KeyRoom-S15` problem (Fig. 8a), the agent needs to fetch a key in a large room (15×15 grids) to enter a smaller room and find a staircase to exit the dungeon. We use the default action space but enable autopickup and therefore remove the `PICKUP` action. We use an episode length of 600 time steps, which is 1.5 times longer than the default episode length.

`KeyChest` is a novel environment designed by us, based on `KeyCorridorS4R3` from MiniGrid (Chevalier-Boisvert et al., 2024) (see Fig. 8b). The agent starts in a corridor randomly connected to different rooms. A key is hidden in one room and a chest in another room. The goal is to open the chest with the key in the inventory. Object positions are randomized. The action space for this task contains 5 discrete actions for moving the agent in 4 cardinal directions (`UP`, `RIGHT`, `DOWN`, `LEFT`) and an `OPEN`-action to open a chest when standing next to it with a key in the inventory. We enable auto-pickup, so no additional action is needed to pick up the key when stepping on it. We use an episode length of 800 time steps.

Rewards All environments use a sparse reward of $r_t = 1$, which the agent only receives upon accomplishing the task. A small punishment ($r_t = -0.01$) is given, when the agent performs an action that does not alter the screen.

Image remapping Empirically, we found that GPT-4 may encounter problems if we provide the image observations as is. For example, when using the default character in the `KeyRoom-S15` environment (Rogue), GPT-4 sometimes throws content violation errors. We suspect that this is due to the character wearing a helmet with horns, which could be mistaken for demonic or satanic imagery. Thus, we pre-processed the images before returning them from the environment. While render all characters as the Tourists, a friendly looking character with a Hawaiian shirt and straw hat. Furthermore, GPT-4 sometimes mistakes entrance staircases for exit staircases. Since the entrance staircases serve no particular purpose and are not different from the regular floor, we remap all entrance staircases to floors.

C.3 VLM prompting

We prompt the VLM with somewhat general descriptions of the environments that we consider. Here we provide the full prompts for all environments.

C.3.1 Robodesk

For Robodesk we provide two observation images (resolution 224×224) with the following prompt:

```
Here are two images in a simulated environment with a robot in front of a desk.
Your task is to pick between these images based on how interesting they are.
Which image is more interesting in terms of the showcased behavior? For context
following points would constitute interestingness: (1) The robot is currently
holding an object in its gripper. (2) The robot is pushing an object around
or pushing a button or opening the drawer or interacting with entities on the
desk. (3) Objects on the desk are in an interesting configuration: e.g. a stack.
Being far away from the desk with the robot arm retracted or just stretching your
arm without interactions, is a sign the image is not interesting. Answer in
maximum one word: 0 for image 1, 1 for image 2, 2 for both images and 3 if you
have no clue.
```

For Robodesk we use GPT-4 Turbo. To cap the dataset size of pairs at 200K and still make sure that we got contrastive pairs of interesting vs. uninteresting behaviors, we pre-filtered the pairs using our oracle of interestingness for the CEE-US and Plan2Explore datasets. We believe that in future work this step can be omitted by simply generating more random pairs and expanding the dataset size.

C.3.2 MiniHack

For MiniHack we provide two observation images (resolution 80×80) and the inventory. For the inventory, we only consider items that were picked up and not items in the agent's starting equipment[3]. We provide the inventory as text descriptions. The different options are shown in purple.

```
Your task is to help play the video game MiniHack. MiniHack is a roguelike
game where an agent needs to navigate through rooms and escape a dungeon. For
succeeding in the game, finding items, collecting items and exploring new rooms is
crucial. Images are egocentric around the agent, who is standing on a dotted blue
floor. Your task is to pick between two game states, composed of images and an
inventory descriptions, based on how interesting and useful they are.

Is there any difference between the first and second game state in terms of how
interesting it is? The images depict the current view. {The first agent has a
key named The Master Key of Thievery in their inventory., The second agent has a
key named The Master Key of Thievery in their inventory., Both agents have a key
named The Master Key of Thievery in their inventory., Both agents have no items in
their inventory.},

Think it through and then answer in maximum one word: 0 if the first state is
more interesting, 1 if the second state is more interesting, 2 if both states are
interesting and 3 if nothing is interesting or you are very unsure.
```

For MiniHack we use GPT-4 omni.

C.4 Oracle for Interestingness

In Robodesk, we use an oracle of interestingness to annotate the pairs. Our goal here is to showcase an upper-bound of performance on SENSEI without the noisiness of VLMs. For the oracle, we deem

[3]The starting equipment is taken from the NetHack game and irrelevant and inaccesible in our tasks.

a state interesting if: (1) any one of the entities are in motion (here only for the ball we make an exception that the ball should be in motion with the end effector close to it as the ball in the environment is unimpeded by friction), (2) if the drawer is opened, (3) if the drawer/sliding cabinet is not yet in motion, but the end effector is very close to their handles, (4) if the upright and flat blocks are not yet in motion but the end effector is very close to them (almost touching), (5) if the stacking task is solved. With these statements, we essentially cover the range of tasks defined in the Robodesk environment, as they are shown in Fig. 9.

D Extended Results

D.1 Robodesk: Rewards

In addition to interaction metrics, we count the number of times task rewards are collected during exploration. Especially with MOTIF distilled from the oracle annotator using CEE-US data, we observe that the agent already solves many tasks in the environment during play. Note that for the `open_slide` task you need to open the slide fully in one direction, which is achieved in abundance in Plan2Explore runs by simply stretching the arm. The full interaction metrics of exploring how the slide moves left-right is not necessarily reflected in the task rewards, as can be seen in comparison to Fig. 4. Additionally as the bin is not really visible in our camera angle, solving `in_bin` tasks are more due to the objects that go off the table landing by chance in the bin for all methods such that higher statistics for `off_table` rewards also lead to higher `in_bin` rewards.

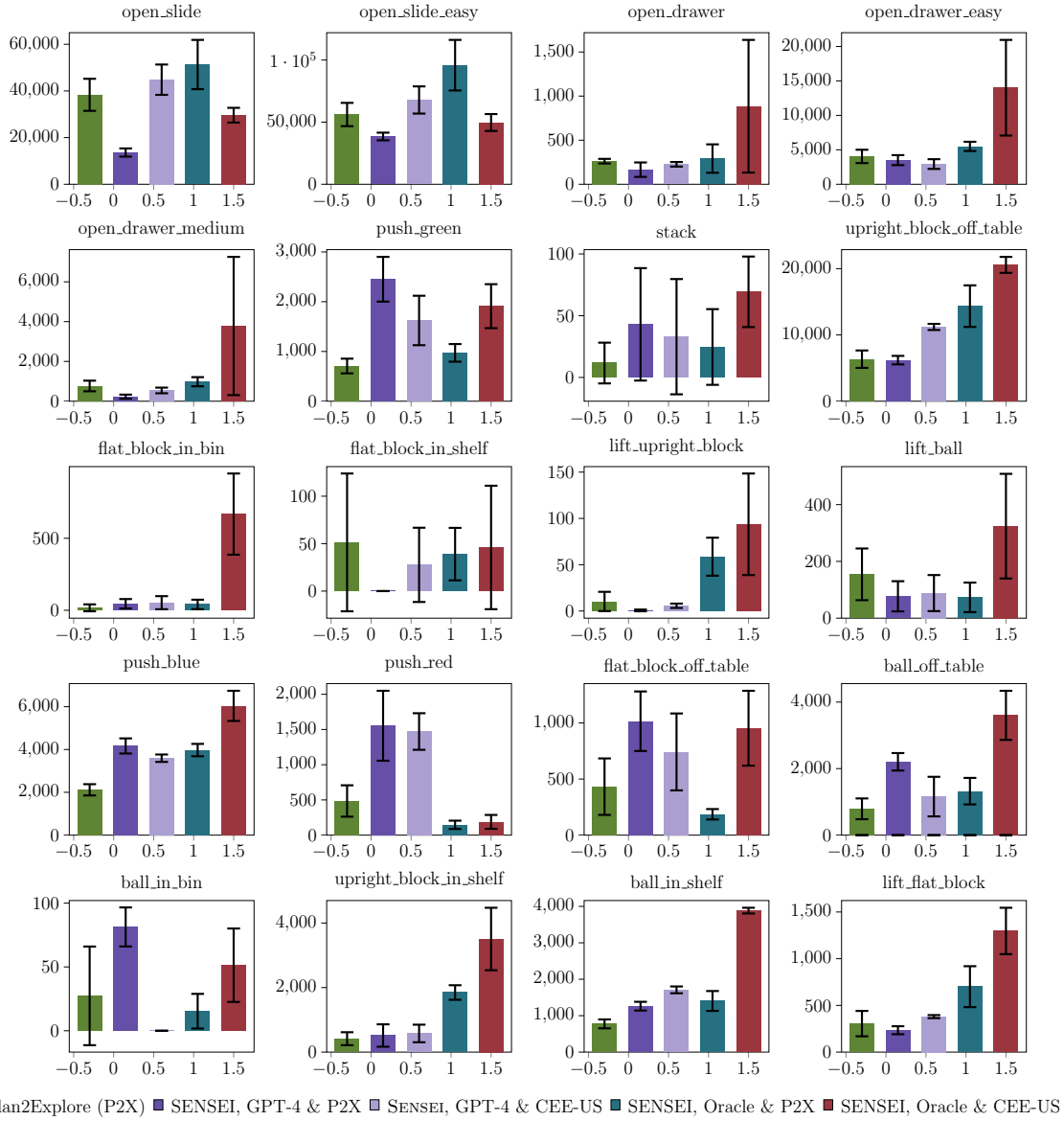


Figure 9: **Robodesk rewards**: We plot the mean number of sparse rewards (successful task completions) discovered during 1M steps of task-free exploration for all tasks.