Check for updates

# Inference of affordances and active motor control in simulated agents

Fedor Scholz[1]*, Christian Gumbsch[1,2], Sebastian Otte[1] and Martin V. Butz[1]

[1]Neuro-Cognitive Modeling Group, Department of Computer Science and Psychology, Eberhard Karls University of Tübingen, Tübingen, Germany, [2]Autonomous Learning Group, Max Planck Institute for Intelligent Systems, Tübingen, Germany

Flexible, goal-directed behavior is a fundamental aspect of human life. Based on the free energy minimization principle, the theory of active inference formalizes the generation of such behavior from a computational neuroscience perspective. Based on the theory, we introduce an output-probabilistic, temporally predictive, modular artificial neural network architecture, which processes sensorimotor information, infers behavior-relevant aspects of its world, and invokes highly flexible, goal-directed behavior. We show that our architecture, which is trained end-to-end to minimize an approximation of free energy, develops latent states that can be interpreted as affordance maps. That is, the emerging latent states signal which actions lead to which effects dependent on the local context. In combination with active inference, we show that flexible, goal-directed behavior can be invoked, incorporating the emerging affordance maps. As a result, our simulated agent flexibly steers through continuous spaces, avoids collisions with obstacles, and prefers pathways that lead to the goal with high certainty. Additionally, we show that the learned agent is highly suitable for zero-shot generalization across environments: After training the agent in a handful of fixed environments with obstacles and other terrains affecting its behavior, it performs similarly well in procedurally generated environments containing different amounts of obstacles and terrains of various sizes at different locations.

KEYWORDS

affordances, active inference, goal-directed control, simulation, free energy principle, model-predictive control, cognitive maps, event-predictive cognition

## 1. Introduction

We, as humans, direct our actions toward goals. But how do we select goals and how do we reach them? In this study, we will focus on a more specific version of the latter question: Given a goal and some information about the environment, how can suitable actions be inferred that ultimately lead to the goal with high certainty?

The free energy principle proposed in Friston (2005) serves as a good starting point for an answer. It is sometimes regarded as a "unified theory of the brain" (Friston, 2010) because it attempts to explain a variety of brain processes such as perception, learning, and goal-directed action selection, based on a single objective: to minimize *free energy*. Free energy constitutes an upper bound on surprise, which results from interactions with the environment. When actions are selected in this way, we also refer to it as active inference. Active inference basically states that agents infer suitable actions by minimizing expected free energy, leading to goal-directed planning.

One limitation of active inference-based planning is computational complexity: Optimal active inference requires an agent to predict the free energy for all possible action sequences potentially far into the future. This soon becomes computationally intractable, which is why so far mostly simple, discrete environments with small state and action spaces have been investigated (Friston et al., 2015). How do biological agents, such as humans, deal with this computational explosion when planning behavior in our complex, dynamic world? It appears that humans, and other animals, have developed a variety of inductive biases that facilitate processing high-dimensional sensorimotor information in familiar situations (Butz, 2008; Butz et al., 2021). *Affordances* (Gibson, 1986), e.g., encode object- and situation-specific action possibilities. By equipping an active inference agent with the tendency to infer affordances, then, inference-based planning could first focus on afforded environmental interactions, significantly alleviating the computational load when considering interaction options.

In this study, we model these conjectures by means of an output-probabilistic, temporally predictive artificial neural network architecture. The architecture is designed to focus on local environmental properties, from which it predicts action-dependent interaction consequences *via* latent state encodings. We show that, through this processing pipeline, *affordance maps* emerge, which encode behavior-relevant properties of the environment. These affordance maps can then be employed during goal-directed planning. Given spatially local visual information, the resulting latent affordance codes constrain the considered environmental interactions. As a result, planning *via* active inference becomes more effective and enables, e.g., the avoidance of uncertainty while moving toward a given goal location. We, furthermore, show that the architecture exhibits zero-shot learning abilities (Eppe et al., 2022), directly solving related environments and tasks within.

# 2. Foundations

This section introduces the theoretical foundations of our study. We first specify our problem setting and notation. We then introduce the free energy principle and show how we can perform active inference-based goal-directed planning with two different algorithms. Subsequently, we combine the theory of affordances with the idea of cognitive maps and arrive at the concept of affordance maps. We propose that the incorporation of affordance maps can facilitate goal-directed planning *via* active inference.

## 2.1. Problem formulation and notation

We consider problems in which an agent interacts with its environment by performing actions $\mathbf{a}$ and in turn receiving sensory states $\mathbf{s}$. The sensory states might reveal only parts of the environmental states $\vartheta$, which, therefore, are not directly observable, i.e., we are facing a partially observable Markov decision process [1]. In every time step $t$, an agent selects and performs an action $\mathbf{a}^t$, and receives a sensory state (often called observation) of the next time step $\mathbf{s}^{t+1}$.

Model-based planning, such as active inference, requires a model of the world to simulate actions and their consequences. We use a *transition model* $t_M$ that predicts the unfolding motor-activity-dependent sensory dynamics while an agent interacts with its environment. In order to deal with partial observability, the transition model can be equipped with its own internal hidden state $\mathbf{h}^t$. Its purpose is to encode the state of the environment, including potentially non-observable parts. Given a current sensory state $\mathbf{s}^t$, an internal hidden state $\mathbf{h}^t$, and an action $\mathbf{a}^t$, the transition model computes an estimate of the sensory state $\tilde{\mathbf{s}}^{t+1}$ in the next time step and a corresponding new hidden state $\mathbf{h}^{t+1}$:

$$(\tilde{\mathbf{s}}^{t+1}, \mathbf{h}^{t+1}) = t_M(\mathbf{s}^t, \mathbf{h}^t, \mathbf{a}^t) \qquad (1)$$

Refer to Figure 1 for a depiction of how environment, agent, transition model, and action selection relate to each other.

## 2.2. Toward free energy-based planning

The *free energy principle* starts formalizing life itself, very generally, as having an interior and exterior, separated by some boundary (Friston, 2013). For life to maintain homeostasis, this boundary, protecting the interior, needs to be maintained. It follows that living things need to be in specific states because only a small number of all possible states ensure homeostasis. The free energy principle formalizes this maintenance of homeostatic states by means of minimizing entropy. But how can entropy be computed? One possibility is given by the presence of an internal, generative model $m$ of the world. In

---

1   In Markov decision processes, usually, the environment additionally returns a reward in each time step, which is to be maximized by the agent. Here, we do not define a reward function but instead plan in a model-predictive, goal-directed manner.

**FIGURE 1**
Depiction of a (partially observable) Markov decision process. An agent interacts with its environment by sending actions **a** and receiving consequent sensory states **s**. *Partial observability* here implies that the sensory state **s** does not encode the whole environmental state $\vartheta$. Rather, certain aspects remain hidden for the agent and must be inferred from the sensory state. To deal with this, our agent utilizes a transition model $t_M$ with its own internal hidden state **h**. It predicts sensory states $\tilde{\mathbf{s}}$, which aid the action selection algorithm to produce appropriate actions. In order to stay in tune with the environment and to predict multiple time steps into the future, the transition model also receives observed and predicted sensory states (dashed arrows).

this case, we can regard entropy as the expected surprise about encountered sensory states given the model (Friston, 2010). In other words: Living things must minimize expected surprises.

This implies that all living things act as if they strive to maintain a model of their environment over time in some way or another. Surprise, however, is not directly accessible to a living thing. In order to compute the surprise corresponding to some sensory input **s**, it is necessary to integrate over all possible environmental states $\vartheta$ that could have led to that input (Friston, 2009). We can see this in the formal definition of surprise for a sensory state **s** (Friston et al., 2010):

$$- \log p(\mathbf{s} \mid m) = - \log \int_{\vartheta} p(\mathbf{s}, \vartheta \mid m) \, d\vartheta \qquad (2)$$

where $m$ is the model or the living thing itself, and $\vartheta$ are all environmental states, including states that are not fully observable to the living thing. The consideration of all these states is infeasible. Thus, according to the free energy principle, living things minimize *free energy*, which is defined as follows (Friston et al., 2010):

$$FE(\mathbf{s}, \mathbf{h}) = \underbrace{E_{q(\vartheta \mid \mathbf{h})}[-\log p(\mathbf{s}, \vartheta \mid m)]}_{\textbf{energy}} - \underbrace{E_{q(\vartheta \mid \mathbf{h})}[-\log q(\vartheta \mid \mathbf{h})]}_{\textbf{entropy}}$$

$$(3)$$

where $E$ denotes the expected value and $q$ is an approximate posterior over the external hidden state $\vartheta$ given internal hidden state **h**. Since here all parameters are accessible, this quantity is computable. Rewriting it shows that free energy can be

decomposed into a surprise and a divergence term:

$$FE(\mathbf{s}, \mathbf{h}) = \underbrace{-\log p(\mathbf{s} \mid m)}_{\textbf{surprise}} + \underbrace{D[q(\vartheta \mid \mathbf{h}) \mid\mid p(\vartheta \mid \mathbf{s}, m)]}_{\textbf{divergence}} \qquad (4)$$

where $D$ denotes the Kullback-Leibler divergence. Since the divergence cannot be less than zero, free energy is an upper bound on surprise, our original quantity of interest.

Given a generative model of the world, surprise corresponds to an unexpected, inaccurate prediction of sensory information. In order to minimize free energy, an agent equipped with a generative world model, thus, has two ways to minimize the discrepancy between predicted and actually encountered sensory information: (i) The internal world model can be adjusted to better resemble the world. In the short term, this relates to *perception*, while in the long term, this corresponds to *learning*. (ii) The agent can manipulate the world *via* its *actions*, such that the world better fits its internal model. In this case, an agent chooses actions that minimize expected free energy in the future, pursuing *active inference*.

### 2.2.1. Active inference

When the free energy principle is employed as a process theory for action selection, it is called *active inference*. The name comes from the fact that the brain actively samples the world to perform inference: It infers actions (also called control states) that minimize expected free energy (EFE), i.e., an upper bound on surprise in anticipated future states. This is closely related to the principle of planning as inference in the machine learning

and control theory communities (Botvinick and Toussaint, 2012; Lenz et al., 2015). According to Friston et al. (2015), a policy $\pi$ is evaluated at time step $t$ by projecting it into the future and evaluating the EFE at some time step $\tau > t$.

$$EFE(\pi, t, \tau) = \underbrace{D[E_{p(\mathbf{h}^\tau | \mathbf{h}^t, \pi)}[p(\mathbf{s}^\tau \mid \mathbf{h}^\tau)] \mid\mid p(\mathbf{s}^\tau \mid m(\tau))]}_{\text{predicted divergence from desired states}}$$
$$+ \beta \cdot \underbrace{E_{p(\mathbf{h}^\tau | \mathbf{h}^t, \pi)}[H[p(\mathbf{s}^\tau \mid \mathbf{h}^\tau)]]}_{\text{predicted uncertainty}}, \quad (5)$$

where $t$ is the current time step, $\tau > t$ is a future time step, and $\beta$ is a new hyperparameter that we introduce. This formula equates EFE with a sum of two components. The first part is the Kullback-Leibler divergence, which estimates how far the predicted sensory states deviate from desired ones. The second part is the entropy of the predicted sensory states, which quantifies uncertainty. We introduce $\beta$ to weigh these components. It enables us to tune the trade-off between choosing actions that minimize uncertainty and actions that minimize divergence from desired states. To calculate the EFE for a whole sequence of $T$ future time steps, we take the mean of the EFE over this sequence:

$$EFE(\pi, t) = \frac{1}{T} \sum_{\tau=t+1}^{t+T} EFE(\pi, t, \tau) \quad (6)$$

Based on this formula, policies can be evaluated and the policy with the least EFE can be chosen:

$$\pi^t = \operatorname*{argmin}_\pi EFE(\pi, t) \quad (7)$$

Intuitively speaking, active inference-based planning agents choose actions that lead to desired sensory states with high certainty.

### 2.2.2. Planning via active inference

On the computational level, active inference tells us to minimize EFE to perform goal-directed planning. Thus, it provides an objective to optimize actions. However, it does not specify how to optimize the actions on an algorithmic level. We, thus, detail two planning algorithms that can be employed for this kind of action selection. In both algorithms, we limit ourselves to a finite prediction horizon $T$ with fixed policy lengths. In order to evaluate policies, both algorithms employ a transition model $t_M$ and "imagine" the execution of a policy:

$$(\tilde{\mathbf{s}}^{\tau+1}, \mathbf{h}^{\tau+1}) = \begin{cases} t_M(\mathbf{s}^t, \mathbf{h}^t, \mathbf{a}^t), & \text{if } \tau = t \\ t_M(\tilde{\mathbf{s}}^\tau, \mathbf{h}^\tau, \mathbf{a}^\tau), & \text{if } t < \tau < t+T \end{cases} \quad (8)$$

For active inference-based planning, we can compute the EFE for the predicted sequence and optimize the actions using one of the planning algorithms. After a fixed number of optimization cycles, both algorithms return a sequence of actions. The first action can then be executed in the environment.

#### Gradient-based active inference

Action inference (Otte et al., 2017; Butz et al., 2019) is a gradient-based optimization algorithm for model-predictive control. Therefore, it requires the transition model $t_M$ to be differentiable. The algorithm maintains a policy $\pi$, which, in each optimization cycle, is fed into the transition model. Afterward, we use backpropagation through time to backpropagate the EFE onto the policy. We obtain the gradient by taking the derivative of the EFE with respect to an action $\mathbf{a}^\tau$ from the policy. After multiple optimization cycles, the algorithm returns the first action of the optimized policy.

#### Evolutionary-based active inference

The cross-entropy method (CEM, Rubinstein, 1999) is an evolutionary optimization algorithm. CEM maintains the parameters of a probability distribution and minimizes the cross-entropy between this distribution and the distribution that minimizes the given objective. It does so by sampling candidates, evaluating them according to EFE, and using the best performing candidates to estimate the parameters of its probability distribution for the next optimization cycle. Recently, CEM has been used as a $zero-order$ optimization technique for model-based control and reinforcement learning (RL) (Chua et al., 2018; Hafner et al., 2019b; Pinneri et al., 2020). In such a model-predictive control setting, CEM maintains a sequence of probability distributions and candidates correspond to policies. After multiple optimization cycles, the algorithm returns the first action of the best sampled policy.

## 2.3. Behavior-oriented predictive encodings

In theory, given a sufficiently accurate model, active inference enables an agent to plan goal-directed behavior regardless of the complexity of the problem. In practice, however, considering all possible actions and consequences thereof quickly becomes computationally intractable. To counteract this problem, it appears that humans and other animals have developed a variety of inductive learning biases to focus the planning process by means of behavior-oriented, internal representations. Here, we focus on biases that lead to the development of affordances, cognitive maps, and, in combination, affordance maps.

### 2.3.1. Affordances

Gibson (1986) defines *affordances* as what the environment offers an animal: Depending on the current environmental

context, affordances are possible interactions. As a result, affordances fundamentally determine how animals behave depending on their environment. They constitute behavioral options from which the animal can select suitable ones in order to fulfill its current goal. To give an example, imagine a flat surface at the height of a human's knees. Given the structure underneath is sufficiently sturdy, it is possible to sit on the surface in a way that requires relatively little effort. Therefore, such a surface is *sit-upon-able*: It offers a human the possibility to sit on it in an effortless way.

In this study, we use a more general definition of affordances. We define affordance as anything in the environment that locally influences the effects of the agent's actions. These definitions differ with respect to the set of possible actions. Gibson's definition entails that certain actions are possible only in certain environmental contexts: For example, sitting down is only possible in the presence of a chair. In this study, we assume that every action is possible everywhere in the environment, but that the effects differ depending on the environmental context: Sitting down is also possible in the absence of a chair, but the effect is certainly different.

The theory of affordances explicitly states that to (visually) perceive the environment is to perceive what it affords. Animals do not see the world as it is and derive their behavioral options from their perspective. Rather, Gibson (1986) proposes that affordances are perceived directly, assigning distinct meanings to different parts of the environment. From an ecological perspective, it appears that vision may have evolved for exactly this purpose: to convey what behaviors are possible in the current situation. First, however, an animal needs to learn the relationship between visual stimuli and their meaning for behavior. This is non-trivial: Similar visual stimuli can mean different things, or the other way round. Furthermore, visual input is rich such that the animal needs to effectively focus on the behavior-relevant information.

### 2.3.2. Cognitive maps

The concept of *cognitive maps* was introduced in Tolman (1948). Tolman showed that after exploring a given maze, rats were able to navigate toward a food source regardless of their starting position. He concluded that the rats acquired a mental representation of the maze: a cognitive map. Place cells in the hippocampus seem to be a promising candidate for the neural correlate of this concept (O'keefe and Nadel, 1978). These cells tend to fire when the animal is at associated locations. Visual input acts as stimuli, but also the olfactory and vestibular senses play a role. Together, place cells constitute a cognitive map, which the animal appears to use for orientation, reflection, and planning (Diba and Buzsaki, 2007; Pfeiffer and Foster, 2013).

### 2.3.3. Affordance maps

Cognitive maps are well-suited for flexible navigation and goal-directed planning. However, to improve the efficiency of the planning mechanisms, it will be useful to encode behavior-relevant aspects, such as the aforementioned affordances, within the cognitive map. Accordingly, we combine the theory of affordances with cognitive maps, leading to *affordance maps*. Their function is to map spatial locations onto affordance codes. Similar to cognitive maps, their encoding depends on visual cues. In contrast to cognitive maps' traditional focus on map-building, though, affordance maps signal distinct behavioral options at particular environmental locations. As an example, consider a hallway corner situation with corridors to your right and behind you. An affordance map would encode successful navigation options for turning to the right or turning around. Regarding affordances for spatial navigation specifically, Bonner and Epstein (2017) showed that these are automatically encoded by the human visual system independent of the current task and propose a location in the brain where a neural correlate could be situated.

## 2.4. Related neural network models

Ha and Schmidhuber (2018) used a world model to facilitate planning *via* RL. Their overall architecture consists of a vision model which compresses visual information, a memory module, and a controller model, which predicts actions given a history of the compressed visual information. Their vision model is given by a variational autoencoder, which is trained in an unsupervised manner to reconstruct its input. Therefore, and in contrast to ours, their vision model is not trained to extract meaningful, behavior-oriented information. Hence, we would not regard the emerging compressed codes as affordance codes.

Affordance maps were used before in Qi et al. (2020) to aid planning. The authors put an agent into an environment (VizDoom) with hazardous regions that were to be avoided. The agent moved around in its environment and collected experiences of harm or no harm, which were backprojected onto the pixels of the input to the agent's visual system, thereby performing image segmentation. The authors then trained a convolutional neural network (CNN) on the resulting data of which the output was utilized by the A* algorithm for planning. In contrast to ours, their architecture was not trained in an end-to-end fashion, meaning that the resulting affordance codes were not optimized to suit their transition model.

## 3. Model

We now detail the proposed architecture, which learns a transition model of the environment with spatial affordance encodings. The architecture predicts a probability density

function over changes in sensory states given the current sensory state and action as well as potentially an internal hidden state. This action-dependent transition model of the environment thus enables active inference-based planning. We first specify the architecture, then detail the model learning mechanism, and finally, turn to active inference.

## 3.1. Affordance-conditioned inference

Our model adheres to the general notion introduced above (cf. Equation 1). Our model consists of three main components: a transition model $t_M$, a vision model $v_M$, and a look-up map $\omega$ of the environment. The model with its different components is illustrated in Figure 2.

Our system learns a transition model $t_M$ of its environment. It receives a current sensory state $\mathbf{s}^t$ and action $\mathbf{a}^t$ and predicts a consequent sensory state $\tilde{\mathbf{s}}^{t+1}$. If the environment is only partially observable, the transition model can furthermore receive an internal hidden state $\mathbf{h}^t$ and predict a consequent $\mathbf{h}^{t+1}$. Focusing on motion control tasks, we encode the sensory state by a two-dimensional positional encoding $\mathbf{p}^t$, where the transition model continues to predict changes in positions $\Delta\tilde{\mathbf{p}}^{t+1}$ given the last positional change $\Delta\mathbf{p}^t$, potentially hidden state $\mathbf{h}^t$, and current action $\mathbf{a}^t$. To enable the model to consider the properties of different regions in an environment during goal-directed planning, though, we introduce an additional contextual input $\mathbf{c}^t$, which is able to modify the transition model's predictions (cf. Butz et al., 2019, for a related approach without map-specificity). In each time step, the transition model $t_M$ thus additionally receives a context encoding vector $\mathbf{c}^t$, which should encode the locally behavior-relevant characteristics of the environment.

This context code is produced by the *vision model* $v_M$, which receives a visual representation $\mathbf{v}^t$ of the agent's current surroundings in the form of a small pixel image. The vision model is, thus, designed to generate vector embeddings that accurately modify the transition model's predictions context-dependently.

The prediction of the transition model can, thus, be formalized as follows:

$$(\Delta\tilde{\mathbf{p}}^{t+1}, \mathbf{h}^{t+1}) = t_M(\mathbf{c}^t = v_M(\mathbf{v}^t), \Delta\mathbf{p}^t, \mathbf{h}^t, \mathbf{a}^t) \qquad (9)$$

The visual information $\mathbf{v}^t$ can be understood as a local view of the environment surrounding the agent. Thus, $\mathbf{v}^t$ depends on the agent's location $\mathbf{p}^t$. To enable the model to predict $\mathbf{v}^t$ for various agent positions, e.g., for "imagined" trajectories while planning, the system is equipped with a *look-up map* $\omega$ to translate positions $\mathbf{p}^t$ into local views of the environment $\mathbf{v}^t$. We augment the model with the ability to probe particular map locations, translate the location into a local image, and extract behavior-relevant information from the image. Intuitively speaking, this

is as if the network can put its focus of attention on any location on the map and consider the context-dependent behavioral consequences at the considered location. As a result, the system is able to consider behavioral consequences dependent on probed environmental locations. In the future study, the learning of completely internal maps may be investigated further.

The consequence of this model design is that the context code $\mathbf{c}^t$ will tend to encode local, behavior-influencing aspects of the environment, i.e., affordances. The context is, therefore, a compressed version of the environment's behaviorally-relevant characteristics at the corresponding position. Therefore, the incorporation of the affordance codes can be expected to improve both the accuracy of action-dependent predictions and active inference-based planning. This connection between active inference and affordances can further be described as follows (Friston et al., 2012): The desired sensory state encoded as a prior lets the agent expect to reach the target. If the agent then is in front of an obstacle e.g., different affordances compete with each other, which is in line with the affordance competition hypothesis (Cisek, 2007): fly around or crash into the obstacle. Since flying around the obstacle best explains the sensory input in light of the prior, the action corresponding to this affordance is chosen by the agent.

## 3.2. Uncertainty estimation

The free energy principle is inherently probabilistic and therefore active inference requires our architecture to produce probability density functions over sensory states. We implement this in terms of a transition model $t_M$ that does not predict a point estimate of the change in sensory state in the next time step, but rather the parameters of a probability distribution over this quantity. We choose the multivariate normal distribution with a diagonal covariance matrix (i.e., covariances are set to 0). The output of the transition model is then given by a mean vector $\mu_{\Delta\tilde{\mathbf{p}}^{t+1}}$ and a vector of standard deviations $\sigma_{\Delta\tilde{\mathbf{p}}^{t+1}}$. We, thus, replace $\Delta\tilde{\mathbf{p}}^{t+1}$ with $\theta_{\Delta\tilde{\mathbf{p}}^{t+1}} := (\mu_{\Delta\tilde{\mathbf{p}}^{t+1}}, \sigma_{\Delta\tilde{\mathbf{p}}^{t+1}})$.

## 3.3. Training

We train both components of our architecture jointly in an end-to-end, self-supervised fashion to perform one-step ahead predictions on a pregenerated data set *via* backpropagation through time. The gradient flow during training is depicted in Figure 2. Inputs consist of the sensor-action tuples described above. The only induced target is given by the change in position in the next time step $\Delta\mathbf{p}^{t+1}$. This target signal is compared to the output of the transition model $t_M$ by

FIGURE 2
Affordance map architecture: Based on the current position $\mathbf{p}^t$, the architecture performs a look-up in an environmental map $\omega$. The vision model $v_M$ receives the resulting visual information $\mathbf{v}^t$ and produces a contextual code $\mathbf{c}^t$. The transition model $t_M$ utilizes this context $\mathbf{c}^t$, the last change in the position $\Delta \mathbf{p}^t$, an action $\mathbf{a}^t$, and its internal hidden state $\mathbf{h}^t$ to predict a probability distribution over the next change in position. During training, the loss between predicted and actual change in position is backpropagated onto $t_M$ (red arrows) and further onto $v_M$ (orange arrows) to train both models end-to-end. During planning, the map look-up is performed using position predictions. For gradient-based active inference, EFE is backpropagated onto the action code $\mathbf{a}^t$ (red and green arrows). For planning with the cross-entropy method, $\mathbf{a}^t$ is modified directly via evolutionary optimization.

the negative log-likelihood (NLL)[2], which approximates free energy assuming no uncertainty in our point estimate $\mathbf{h}$ of environmental state $\vartheta$ (refer to Equation 1). Due to end-to-end backpropagation, the vision model $v_M$ is trained to output compact, transition model-conditioning representations of the visual input.

While we use the NLL as the objective during training here, we make use of the expected free energy during goal-directed control. In future study, one could utilize full-blown FE also during training in probabilistic architecture. However, there is a close relationship between NLL and FE due to the Kullback-Leibler divergence: In Appendix 6, we show that minimizing NLL is equivalent to minimizing the Kullback-Leibler divergence up to a constant factor and a constant. Thus, through NLL-based learning, we can approximate learning through FE minimization.

## 3.4. Goal-directed control

We perform goal-directed control via gradient- and evolutionary-based active inference as described in Section 2.2.1. Usually, in order to predict multiple time steps into the future

given a policy, the transition model $t_M$ receives its own output as input. Since our architecture predicts the parameters of a normal distribution, we use the predicted mean as input in the next prediction time step. The model incorporates visual information $\mathbf{v}$ from locations corresponding to the predicted means. Therefore, the model does not blindly imagine a path but simultaneously "looks" at, or focuses on, predicted positions, incorporating the inferred affordance code $\mathbf{c}$ into the transition model's predictions.

In order to compare the predicted path to the given target and to look up the visual information, we need absolute locations. We, thus, take the cumulative sum and add the current absolute position. To compute EFE along a predicted path we also need to consider the SDs at every point. For that, we first convert SDs to variances, compute the cumulative sum, and convert them back to SDs. We then can compute the EFE between the resulting sequence of probability distributions over predicted absolute positions and the given target according to Equation (5). To do so, we encode the target with a multivariate normal distribution as well, setting the mean to the target location and the SD to a fixed value. We can, thus, optimize the policy via the gradient- or evolutionary-based EFE minimization method introduced in Section 2.2.2 above[3].

---

2    Refer to Appendix 4 for a description of how to compute gradients when the objective is given by the NLL in a multivariate normal distribution.

3    Appendix 2 summarizes the particular adjustments we applied to these algorithms.

**FIGURE 3**
The simulation environment we use in our experiments. It resembles a confined space in which a vehicle (green) can move around by adjusting its throttles (blue). Its goal is to navigate toward the target (red). Depending on the experiment, obstacles or different terrains (black) are present, which affect the vehicle's sensorimotor dynamics.
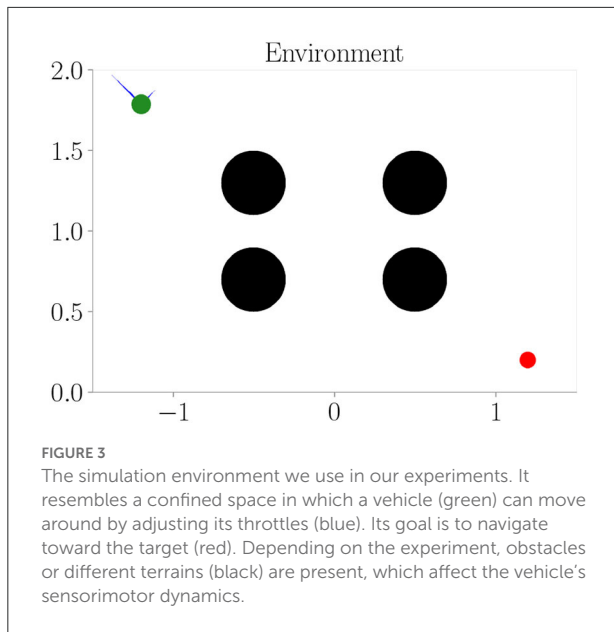
# 4. Experiments and results

To evaluate the abilities of our neural affordance map architecture, we first introduce the environmental simulator and specify our evaluation procedure in general. The individual experimental results then evaluate the system's planning abilities to avoid obstacles and regional uncertainty as well as to generalize to unseen environments. With respect to the affordance codes **c**, we show emerging affordance maps and examine disentanglement.

## 4.1. Environment

The environment used in our experiments is a physics-based simulation of a circular, vehicle-like agent with a radius of 0.06 in a 2-dimensional space with an arbitrary size of 3 × 2 units. It is confined by borders, which prevent the vehicle from leaving the area. The vehicle is able to fly around in the environment by adjusting its 4 throttles, which are attached between the vertical and horizontal axes in a diagonal fashion. They take values between 0 and 1 resembling actions and enable the vehicle to reach a maximum velocity of approximately 0.23 units per time step within the environment. Therefore, at least 13 time steps are required for the vehicle to fly from the very left to the very right. Due to its mass, the vehicle undergoes inertia and by default, it is not affected by gravity. Refer to Figure 3 for a depiction of the environment and an agent. It is implemented as an OpenAI Gym (Brockman et al., 2016).

The environment can contain obstacles, which block the way. Friction values are larger when the vehicle touches obstacles

or borders. Furthermore, the environment can comprise different terrains, which locally change the sensorimotor dynamics. Force fields pull the agent up or downward. If the vehicle is inside a fog terrain, the environment returns a position that is corrupted by Gaussian noise. Two values from a standard normal distribution are sampled and added to each coordinate. This implies a SD of approximately 1.414 on the difference between positions from two consecutive time steps within the fog[4].

The environment outputs absolute positions, the change in the positions, and allows probing of the map at arbitrary positions. Therefore, and apart from the noisy positions in fog terrains and the map having a lower resolution than the environment itself, the environment used in our experiments is fully observable. This makes the incorporation of the internal hidden state **h** in the transition model $t_M$ obsolete. In the future, we plan to test our architecture on environments that are only partially observable.

## 4.2. Model and agent

The vision model $v_M$ is given by a CNN, which produces the context activations **c**. We always evaluate contexts of sizes 0, 1, 3, 5, and 8. Since the environment in our experiments is almost fully observable, we drop the internal hidden state **h** and use a multilayer perceptron (MLP) as our transition model $t_M$. It consists of a fully connected layer followed by two parallel fully connected layers for the means and standard deviations, respectively. For each setting, we train 25 versions of the same architecture with different initial weights on a pregenerated data set and report aggregated results. Refer to Appendix 2 for more details on the model and training hyperparameters, how the visual input **v** is constructed, how the data set is generated, and the training procedure.

Active inference performance is evaluated after performing 100 goal-directed control runs per setting for 200 time steps. For each trained architecture instance, we consider 4 distinct start and target positions corresponding to each corner of the environment. The start position is chosen randomly with a uniform distribution over a $0.2 \times 0.2$ units square with a distance of 0.1 units to the borders. The target position is chosen in the same way in the diagonally opposite corner. We consider the agent to have reached the target when its distance to the target falls below 0.1 units. The prediction horizon always has a length of 20. For active inference based planning (Equation 5), the target is provided to the system as a Gaussian distribution with SD 0.1. We reduce the SD of the target distribution to 0.01 once the agent comes closer than 0.5 units. The two different SDs can be seen as corresponding to, e.g., smelling and seeing

---

4   Since $\sqrt{1^2 + 1^2} = \sqrt{2} \approx 1.414$.

the target, respectively. Refer to Appendix 2 for more details on the hyperparameters. All hyperparameters were optimized empirically or with Hyperopt (Bergstra et al., 2013) *via* Tune (Liaw et al., 2018).

In order to get an idea about the nature of the emerging affordance codes **c**, we plot affordance maps by generating position-dependent context activations *via* the vision model $v_M$ for each possible location in the environment. That is, in the affordance maps shown below, the x- and y-axes correspond to locations in the environment while the color of each dot represents the context activation at that position. For this, we performed a principal component analysis (PCA) on the resulting context activations in order to reduce the dimensionality to 3 and then interpreted the results as RGB values.

## 4.3. Experiment I: Obstacle avoidance

The first of our experiments examine our architecture's ability to avoid obstacles during active inference through the use of affordance codes. As a baseline experiment, we consider context size 0, disabling information flow from the vision model $v_M$ to the transition model $t_M$. With context sizes larger than 0, however, the transition model can be informed about obstacles and borders *via* the context.

We train the architecture on the environment depicted in Figure 3, where black areas resemble obstacles. One-hot encoded visual information **v** has one channel only for the obstacles and borders. We perform goal-directed control in the same environment we train.

Figure 4 shows the results. Context codes of increasing dimensionality lead to smaller validation losses (Figure 4A), indicating their utility in improving the transition model's accuracy. The affordance map (Figure 4B) shows that obstacles are encoded differently from the rest of the environment. Areas where free flight if possible are encoded with a context code that corresponds to olive green. In contrast, areas where it is only possible to fly upward, to the left, or downward e.g., are encoded with a context code that corresponds to the color orange. Light green, on the other hand, represents areas where the only movement to the left is blocked. The affordance map reveals that different sides of the obstacles are encoded similarly to the corresponding sides of the environment's boundary. Moreover, we find gradients in the colors when moving away from borders or obstacles, indicating that the context codes not only encode directions but also distances to impassable areas. This confirms that the emerging context codes constitute behavior-relevant encodings of the visually perceived environment. We evaluate goal-directed planning in terms of prediction error (Figure 4C) and mean distance to the target (Figure 4D). For evolutionary-based active inference, we find improvement in both metrics with

increasing context sizes. For gradient-based active inference, we find improvement in the prediction error but deterioration of the mean distance to the target with increasing context sizes. Gradient-based outperforms evolutionary-based active inference with context size 0. With larger context sizes, evolutionary-based active inference performs better. Figure 5 shows two example trajectories for context sizes 0 and 8. A context size of 8 allows the agent to incorporate local information about the environment, resulting in past and planned trajectories that bend around obstacles. With context size 0, however, it can be seen that the agent flew against one obstacle and plans its trajectory through another one.

## 4.4. Experiment II: Generalization

In this experiment, we examine how well our architecture is able to generalize to similar environments. In Experiment I (Section 4.3), we trained in a single environment. Once the architecture is trained, we expect that our system should be able to successfully perform goal-directed control in other environments as well, given we provide the corresponding visual input. The local view on the map essentially allows us to change the position and size of obstacles without expecting a significant deterioration in performance.

We reuse the trained models from Experiment Section 4.3 and apply them for goal-directed control in two additional environments (refer to Figure 6). We only consider evolutionary-based active inference.

Figure 7 shows the results. Prediction error and mean distance to the target (Figures 7C,D) indicate improvement with increasing context size. Furthermore, we find slightly worse performance in the environment with 2 obstacles, while slightly better performance is achieved in the environment with 12 obstacles. We believe this is mainly due to the fact that the environment with 2 obstacles blocks the direct path much more severely. Thus, overall these results indicate that (i) the system generalized well to similar environments and (ii) incorporating context codes is beneficial for performance optimization.

## 4.5. Experiment III: Behavioral-relevance of affordance codes

Affordances should only encode visual information if it is relevant to the behavior of an agent. Is our architecture able to ignore visual information for creating its affordance maps, if this information has no effect on the agent's behavior? Furthermore, affordances should encode different visual information with the same behavioral meaning similarly. To investigate our architecture in this regard, we perform an experiment similar to Experiment I (Section 4.3), but with two additional channels in the cognitive map. The first channel encodes the borders and

FIGURE 4
Results for Experiment I—Obstacle avoidance (Section 4.3). For each setting, the results are aggregated over 25 differently initialized models which each performed 4 goal-directed control runs for 200 time steps. The box plots show the medians (horizontal bars), quartiles (boxes), and minima and maxima (whiskers). Data points outside of the range defined by extending the quartiles by 1.5 times the interquartile range in both directions are ignored. EB is short for evolutionary- and GB for gradient-based active inference. **(A)** Validation loss during training. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. Shaded areas represent SDs. **(B)** Exemplary affordance map for context size 8. To generate this map, we probed the environmental map at every sensible location, applied the vision model to each output, performed dimensionality reduction to 3 *via* PCA, and interpreted the results as RGB values. **(C)** Prediction error during goal-directed control. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. **(D)** Mean distance to the target during goal-directed control.

upper obstacles, the second channel encodes the lower obstacles, and the third channel encodes meaningless information, which does not affect the behavior of the agent. Figure 8 shows the corresponding environment. We compare the results from this "hard condition" to the results of Experiment I (Section 4.3), which we refer to as the "easy condition." We only consider evolutionary-based active inference.

Figure 9 shows the results. We do not find a significant difference between the two conditions. The developing affordance map (Figure 9B) is qualitatively similar to the one obtained from Experiment I (Section 4.3): neither do significant visual differences between the encodings of the different obstacles remain, nor do traces of the meaningless information. Appendix 3 exemplarily shows how this affordance map develops over the course of training. Finally, also performance

in terms of prediction error and mean distance to the target stays similar to Experiment I when analyzing goal-directed control (Figures 9C,D).

## 4.6. Experiment IV: Uncertainty avoidance

Active inference considers uncertainty during goal-directed control. In this experiment, we examine the architecture's ability to avoid regions of uncertainty during planning. We consider a run a success if the agent reached the target and was at no point inside a fog terrain. As mentioned above, we introduce an additional hyperparameter $\beta$, which scales the influence of the entropy term on the free energy (refer to Equation 5). Here, we

**FIGURE 5**
Example trajectories from Experiment I–Obstacle avoidance (Section 4.3 with evolutionary-based active inference. The agent (green) flies toward the target (red) with obstacles (gray) in its way. The black line behind the agent shows its past trajectory and the line in front its planned trajectory. Circles in front of the agent show the predicted uncertainty in the sensory states. With context size 0, the agent cannot incorporate information about the environment and, therefore, plans through and flies against the obstacles. With context size 8, the agent can successfully plan its way around and, therefore, avoid obstacles.



**FIGURE 6**
Additional environments used during goal-directed control in Experiment II–Generalization (Section 4.4). Black areas represent obstacles.

set $\beta$ to 10 to foster avoidance of uncertainty. We only consider evolutionary-based active inference.

We train the architecture on the environment depicted in Figure 3, this time black areas indicate fog terrains instead of obstacles. The cognitive map consists of two channels: one channel for fog terrains and one channel for the borders.

Figure 10 shows the results. We find that the context encoding clearly improves the validation loss (Figure 10A). The affordance map (Figure 10B) shows that free areas, borders, and fog are encoded differently. The prediction error (Figure 10C) improves when the context is computed,

while the ratio of successful runs (Figure 10D) stays relatively close to 1.

## 4.7. Experiment V: Disentanglement

In our final experiment, we examined the architecture's ability to combine previously learned affordance codes. We trained each architecture instance in four different environments. The environments are constructed as shown in Figure 3, black areas resembling obstacles in the first, fog terrains in the second, force fields pointing upward in the third,

**FIGURE 7**
Results from Experiment II–Generalization (Section 4.4). For each setting, the results are aggregated over 25 differently initialized models which each performed 4 goal-directed control runs for 200 time steps. The box plots show the medians (horizontal bars), quartiles (boxes), and minima and maxima (whiskers). Data points outside of the range defined by extending the quartiles by 1.5 times the interquartile range in both directions are ignored. "Original" refers to the environment from Experiment I (Section 4.3). **(A)** Exemplary affordance map for context size 8 from the environment with 2 obstacles. To generate this map, we probed the environmental map at every sensible location, applied the vision model to each output, performed dimensionality reduction to 3 *via* PCA, and interpreted the results as RGB values. **(B)** Exemplary affordance map for context size 8 from the environment with 12 obstacles. **(C)** Prediction error during goal-directed control on all three environments. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. **(D)** Mean distance to the target during goal-directed control in all three environments.
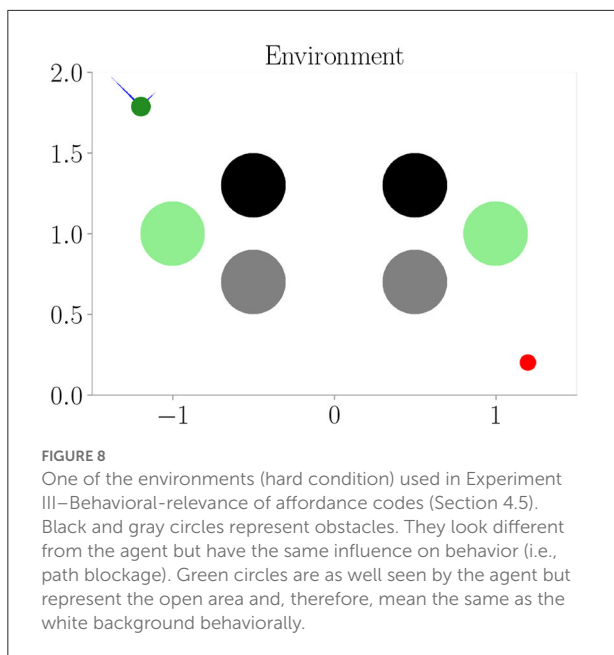
and force fields pointing downward in the fourth environment. Accordingly, the cognitive map consists of four channels—one channel for each of the aforementioned properties. We evaluate the architecture of procedurally generated environments. In each environment, a randomly chosen amount of between 6 and 10 obstacles, fog terrains, force fields pointing downward, and force fields pointing upward with randomly chosen radii between 0.1 and 0.5 are placed at random locations in the environment. All obstacles and fog terrains have a minimum distance of 0.15 units from each other, ensuring that the agent is able to fly between them—thus prohibiting dead ends. Furthermore, all properties have a minimum distance of 0.15 to each border, again to avoid dead ends. Patches of size 0.4 × 0.4 units are left free in the corners such that start and target positions are not affected. We generate environments with two

different conditions. In the first condition (easy), force fields are handled similarly to obstacles and fog terrains in the way that they have a minimum distance of 0.15 units to all other obstacles, terrains, and force fields. This means that properties do not overlap. In the second condition (hard), force fields can overlap with each other, obstacles, and fog terrains. We only consider evolutionary-based active inference. In addition to the context sizes from before, we also evaluate the architecture for context sizes 16 and 32.

Figure 11 shows the results. Larger context sizes lead to smaller validation losses (Figure 11A). The affordance map computed on an environment containing all properties (Figure 11B) shows that the network has learned to encode the distinct areas indeed with distinct encodings. The encoding also incorporates boundary directions, thus encoding the properties

**FIGURE 8**
One of the environments (hard condition) used in Experiment III—Behavioral-relevance of affordance codes (Section 4.5). Black and gray circles represent obstacles. They look different from the agent but have the same influence on behavior (i.e., path blockage). Green circles are as well seen by the agent but represent the open area and, therefore, mean the same as the white background behaviorally.

relative to the free space from which the agent may enter the area (refer to, e.g., the borders of the environment). As expected, the agent always performs better in the easy condition (Figures 11C,D). In both conditions, performance improves with increasing context size.
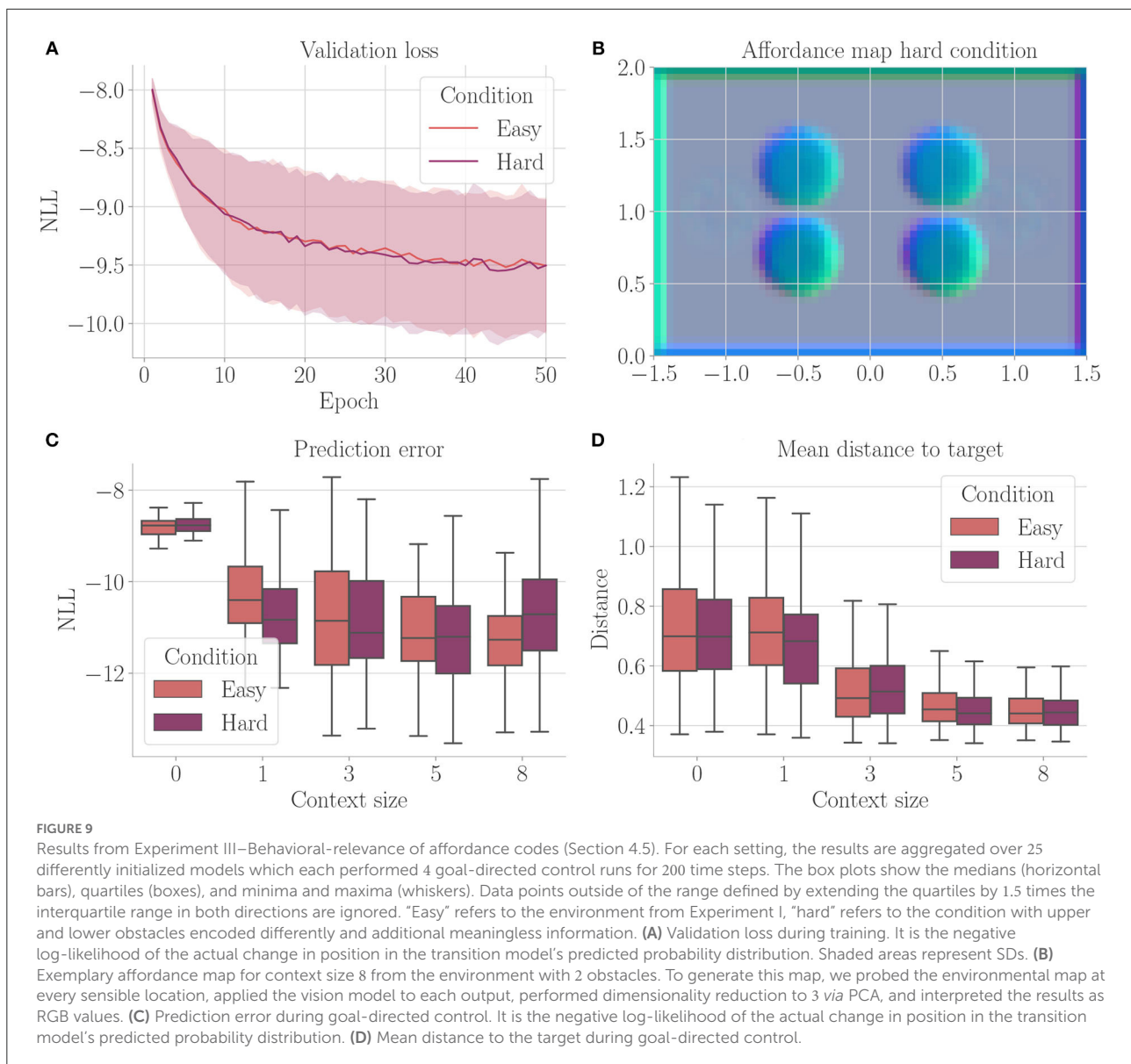
# 5. Discussion

In this article, we have connected active inference with the theory of affordances in order to guide the search for suitable behavioral policies *via* active inference in recurrent neural networks. The resulting architecture is able to perform goal-directed planning while considering the properties of the agent's local environment. This chapter provides a summary of our architecture's abilities, compares it to related study, and eventually presents possible future study directions.

## 5.1. Conclusion

Experiment I (Section 4.3) showed that our proposed architecture facilitates goal-directed planning *via* active inference. Both the validation loss as well as performance during goal-directed control revealed an advantage of incorporating affordance information, i.e., using a context size larger than 0. The affordance maps confirmed that the architecture is able to infer relationships between environmental features and their meaning for the agent's behavior: Depending on the direction of and the distance to the next obstacle, different codes emerged. We assume that a context size larger than 1 allows an easier encoding of the direction of and distance to the obstacles

in relation to the agent. Evolutionary-based active inference outperforms gradient-based active if a context is used. This is due to the fact that gradient-based active inference relies on the gradients being backpropagated through the predicted sequence of sensory states. These gradients cannot be backpropagated through the context codes, since these depend on the visual information which the model acquires *via* a look-up in the environmental map. Therefore, vital information is missing during gradient-based active inference in order to optimize the policy accordingly. Experiment II (Section 4.4) showed that once the relationship between environmental features and their meaning was learned, this knowledge can be generalized to other environments with similar, but differently sized and positioned obstacles of different amounts. Experiment III (Section 4.5) showed that our architecture is able to map properties of the environment that are encoded differently visually but have the same influence on behavior onto the same affordance codes. This matches our general definition of affordances, namely an affordance encoding locally behavior-modifying properties of the environment. In the future, we plan to evaluate our architecture in environments where the connection to task-relevancy is more concrete. An example would be a key that needs to be picked up by the agent in order for a door to be encoded as passable. Experiment IV (Section 4.6) showed that our architecture is able to avoid regions of uncertainty (fog terrains) during planning *via* active inference. Experiment V (Section 4.7) emphasized our architecture's generalization abilities but also showed that the learned affordances are not disentangled. If properties of the environment do not overlap and with a sufficiently large context size, the agent can successfully reach the target without touching regions of uncertainty nearly all the time. This is less so if properties do overlap. We propose that an additional regularization that may foster a disentanglement or factorization of the learned affordances could lead to a fully successful generalization to arbitrary combinations of previously encountered properties.

Our notion of affordances admittedly slightly differs from the original definition in Gibson (1986). In this study, we assume that every action is possible everywhere, but that only the effects differ depending on the environmental context. This was certainly the case in the environment we used in our experiments. We think that this is also often the case in the real world—particularly when actions are considered on the lowest level only, i.e., muscle movements. When increasing behavioral abstraction, though, this might not necessarily be the case any longer. For example, the high-level, composite action of driving a nail into a wall clearly is not possible in every environmental context. In the future, we want to investigate how our architecture can be expanded to flexibly and hierarchically process event-like structures (Zacks and Tversky, 2001; Zacks et al., 2007; Butz et al., 2021; Eppe et al., 2022). In order to foster these event structures, inductive biases as in Butz et al. (2019) and Gumbsch et al. (2021) might be necessary, which assume that most of the time agents are within ongoing events and that

**FIGURE 9**
Results from Experiment III—Behavioral-relevance of affordance codes (Section 4.5). For each setting, the results are aggregated over 25 differently initialized models which each performed 4 goal-directed control runs for 200 time steps. The box plots show the medians (horizontal bars), quartiles (boxes), and minima and maxima (whiskers). Data points outside of the range defined by extending the quartiles by 1.5 times the interquartile range in both directions are ignored. "Easy" refers to the environment from Experiment I, "hard" refers to the condition with upper and lower obstacles encoded differently and additional meaningless information. **(A)** Validation loss during training. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. Shaded areas represent SDs. **(B)** Exemplary affordance map for context size 8 from the environment with 2 obstacles. To generate this map, we probed the environmental map at every sensible location, applied the vision model to each output, performed dimensionality reduction to 3 *via* PCA, and interpreted the results as RGB values. **(C)** Prediction error during goal-directed control. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. **(D)** Mean distance to the target during goal-directed control.
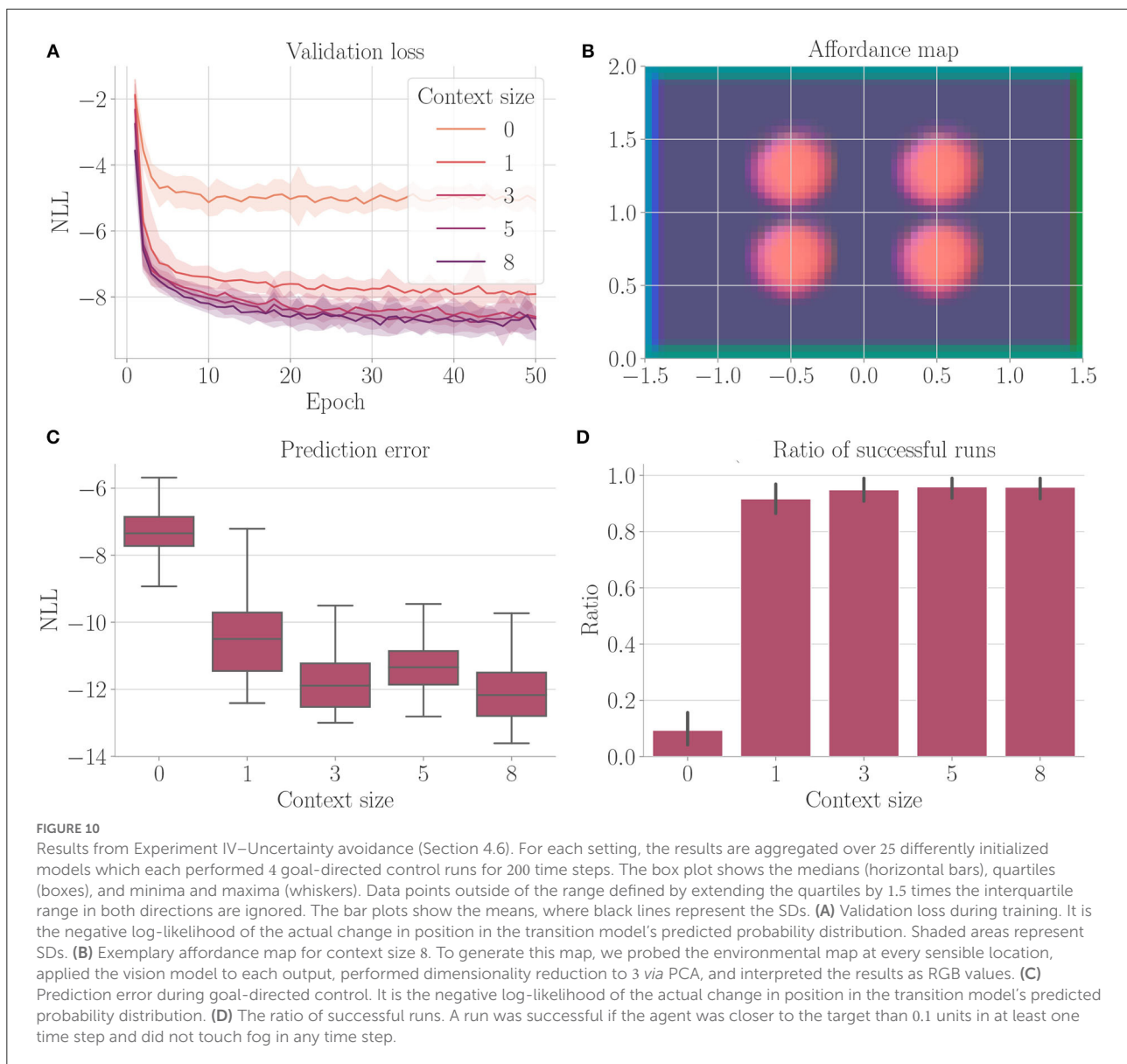
event boundaries characterize transitions between events. Such event models may thus set the general context. The proposed vision model may then be conditioned on this context to enable accurate, event-conditioned action-effect predictions. As a result, the event-conditioned model would learn to encode when the action result that is associated with a particular event-specific affordance can be accomplished.

We treat the context size in our experiments as a hyperparameter, which needs to be set by the experimenter. Our results show that when the context size is too small, the system is unable to learn all environmental influences on action effects. Larger context sizes, on the other hand, tend to decrease prediction error. Improvement does not only depend on the context size, but also the computational capacity of the vision

and transition models. What context size is necessary for a certain number of possible environmental interactions remains an open question for future research. One possible direction here is to let the model adapt the context size on demand. In this case, the computational capacity of the vision and transition models need to be adapted as well, which poses a challenge. Furthermore, the transition model may infer, if equipped with recurrences to deal with partial observability, information that would otherwise be immediately available *via* the vision module. This leads to competition during learning, which needs to be studied further.
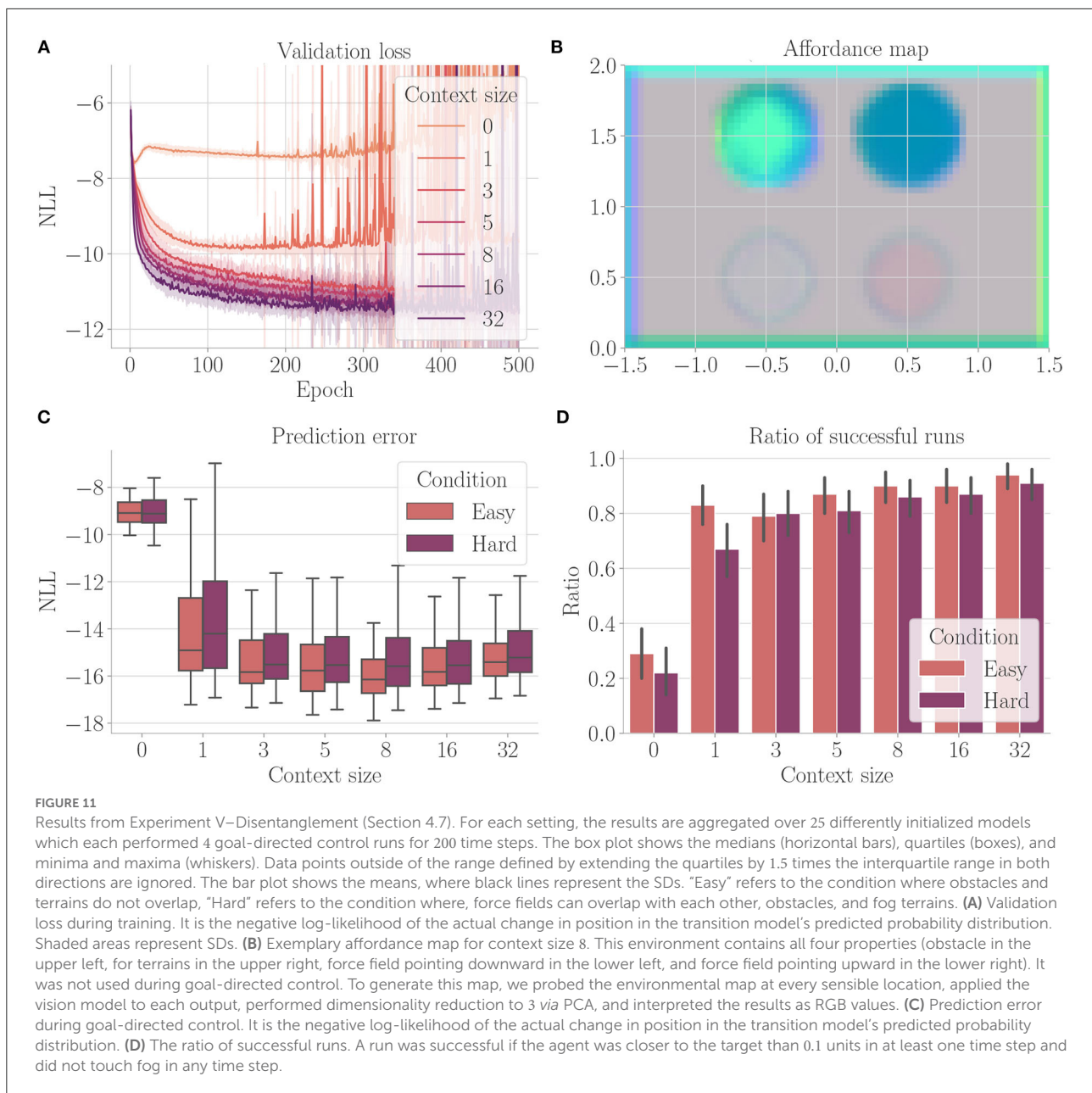
Even though our architecture was successful in solving the presented tasks, it clearly has some limitations. First, our model computes affordance codes directly from visual

FIGURE 10
Results from Experiment IV–Uncertainty avoidance (Section 4.6). For each setting, the results are aggregated over 25 differently initialized models which each performed 4 goal-directed control runs for 200 time steps. The box plot shows the medians (horizontal bars), quartiles (boxes), and minima and maxima (whiskers). Data points outside of the range defined by extending the quartiles by 1.5 times the interquartile range in both directions are ignored. The bar plots show the means, where black lines represent the SDs. **(A)** Validation loss during training. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. Shaded areas represent SDs. **(B)** Exemplary affordance map for context size 8. To generate this map, we probed the environmental map at every sensible location, applied the vision model to each output, performed dimensionality reduction to 3 *via* PCA, and interpreted the results as RGB values. **(C)** Prediction error during goal-directed control. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. **(D)** The ratio of successful runs. A run was successful if the agent was closer to the target than 0.1 units in at least one time step and did not touch fog in any time step.

information. Since it is not able to memorize which affordances are where it constantly needs to perform look-ups on the environmental map. Second, our proposed architecture solves the considered tasks in a greedy manner. During planning, we compare the predicted sensory states to a fixed desired sensory state over the predicted trajectory. Our model, therefore, prefers actions that lead closer to the target only within the prediction horizon. This can be problematic if we consider e.g., tool use. Imagine an environment with keys and doors. Here, it might be necessary to temporarily steer away from the target in order to pick up a key and eventually get closer to the target after unlocking and passing through the door. Without further modifications, our agent would not make such a detour deliberately. In the future study section

below, we make suggestions on how these limitations may be overcome.

Reinforcement learning (RL) (Sutton and Barto, 2018) is another popular approach for solving POMDPs. Therefore, in future study, it would be interesting to see how an RL agent performs in comparison to our agent. A central aspect of our architecture is the look-up in the environment which makes the emergence of affordance maps possible. While certainly possible, it is not straight-forward how this would be implemented in a classical RL agent. Classical RL agents do not predict positional changes which are necessary for the look-up. Furthermore, it was shown that RL agents struggle with offline learning (Levine et al., 2020) and generalization to similar environments (Cobbe et al., 2019).

**FIGURE 11**
Results from Experiment V–Disentanglement (Section 4.7). For each setting, the results are aggregated over 25 differently initialized models which each performed 4 goal-directed control runs for 200 time steps. The box plot shows the medians (horizontal bars), quartiles (boxes), and minima and maxima (whiskers). Data points outside of the range defined by extending the quartiles by 1.5 times the interquartile range in both directions are ignored. The bar plot shows the means, where black lines represent the SDs. "Easy" refers to the condition where obstacles and terrains do not overlap, "Hard" refers to the condition where, force fields can overlap with each other, obstacles, and fog terrains. **(A)** Validation loss during training. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. Shaded areas represent SDs. **(B)** Exemplary affordance map for context size 8. This environment contains all four properties (obstacle in the upper left, for terrains in the upper right, force field pointing downward in the lower left, and force field pointing upward in the lower right). It was not used during goal-directed control. To generate this map, we probed the environmental map at every sensible location, applied the vision model to each output, performed dimensionality reduction to 3 *via* PCA, and interpreted the results as RGB values. **(C)** Prediction error during goal-directed control. It is the negative log-likelihood of the actual change in position in the transition model's predicted probability distribution. **(D)** The ratio of successful runs. A run was successful if the agent was closer to the target than 0.1 units in at least one time step and did not touch fog in any time step.

## 5.2. Future study

In this study, we trained our architecture on previously collected data and only afterward performed goal-directed control. Alternatively, one could perform goal-directed control from the very beginning and train the architecture on inferred actions and the corresponding encountered observations in a self-supervised learning manner. This should increase performance since the distribution of the training data for the transition model then more closely matches the distribution of the data encountered during control. In this case, the exploration-exploitation dilemma needs to be resolved: How

should the agent decide whether it should exploit previously acquired knowledge to reach its goal or instead explore the environment to gain further knowledge that can be exploited in later trials? The active inference mechanism (Friston et al., 2015) generally provides a solution to this problem, although optimal parameter tuning remains challenging (Tani, 2017).

Future study could also examine to what extent it is possible to fully memorize affordances akin to a cognitive map. A straightforward approach would be to train a multi-layer perceptron that maps absolute positions onto affordance codes, in which case translational invariance is lost. Alternatively, a recurrent neural network that receives actions could predict

affordances in future time steps conditioned upon previously encountered affordances. Additionally, the transition model could be split into an encoder, which maps sensory states onto internal hidden states, and a transition model, which maps internal hidden states and actions onto the next internal hidden states. The introduction of an observation model that translates internal hidden states back into sensory states would then enable the whole planning process to take place in a hidden state space akin to PlaNet (Hafner et al., 2019b) and Dreamer (Hafner et al., 2019a).

## Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

## Author contributions

All authors contributed to the conceptualization of the model, which was implemented by FS. All authors contributed to the writing of the manuscript, thereby reading and approving it.

## Funding

## Acknowledgments

## Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

## Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

## Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnbot.2022.881673/full#supplementary-material

## References

Bergstra, J., Yamins, D., and Cox, D. (2013). "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proceedings of the 30th International Conference on Machine Learning*, Vol. 28, eds S. Dasgupta and D. McAllester (Atlanta, GA: PMLR), 115–123. Available online at: http://proceedings.mlr.press/v28/bergstra13.pdf

Bonner, M. F., and Epstein, R. A. (2017). Coding of navigational affordances in the human visual system. *Proc. Natl. Acad. Sci. U.S.A.* 114, 4793–4798. doi: 10.1073/pnas.1618228114

Botvinick, M., and Toussaint, M. (2012). Planning as inference. *Trends Cogn. Sci.* 16, 485–488. doi: 10.1016/j.tics.2012.08.006

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., et al. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*. doi: 10.48550/arXiv.1606.01540

Butz, M. V. (2008). How and why the brain lays the foundations for a conscious self. *Construct. Foundat.* 4, 1–42.

Butz, M. V., Achimova, A., Bilkey, D., and Knott, A. (2021). Event-predictive cognition: a root for conceptual human thought. *Top. Cogn. Sci.* 13, 10–24. doi: 10.1111/tops.12522

Butz, M. V., Bilkey, D., Humaidan, D., Knott, A., and Otte, S. (2019). Learning, planning, and control in a monolithic neural event inference architecture. *arXiv:1809.07412 [cs]*. *arXiv: 1809.07412*. doi: 10.1016/j.neunet.2019.05.001

Chua, K., Calandra, R., McAllister, R., and Levine, S. (2018). "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in Advances in Neural Information Processing Systems, Vol. 31, eds S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Curran Associates). Available online at: https://proceedings.neurips.cc/paper/2018/file/3de568f8597b94bda53149c7d7f5958c-Paper.pdf

Cisek, P. (2007). Cortical mechanisms of action selection: the affordance competition hypothesis. *Philos. Trans. R. Soc. B Biol. Sci.* 362, 1585–1599. doi: 10.1098/rstb.2007.2054

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. (2019). "Quantifying generalization in reinforcement learning," in *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97, eds K. Chaudhuri and R. Salakhutdinov (PMLR), 1282–1289. Available online at: http://proceedings.mlr.press/v97/cobbe19a/cobbe19a.pdf

Diba, K., and Buzsaki, G. (2007). Forward and reverse hippocampal place-cell sequences during ripples. *Nat. Neurosci.* 10, 1241–1242. doi: 10.1038/nn1961

Eppe, M., Gumbsch, C., Kerzel, M., Nguyen, P. D. H., Butz, M. V., and Wermter, S. (2022). Intelligent problem-solving as integrated hierarchical reinforcement learning. *Nat. Mach. Intell.* 4, 11–20. doi: 10.1038/s42256-021-00433-9

Friston, K. (2005). A theory of cortical responses. *Philos. Trans. R. Soc. B Biol. Sci.* 360, 815–836. doi: 10.1098/rstb.2005.1622

Friston, K. (2009). The free-energy principle: a rough guide to the brain? *Trends Cogn. Sci.* 13:293–301. doi: 10.1016/j.tics.2009.04.005

Friston, K. (2010). The free-energy principle: a unified brain theory? *Nat. Rev. Neurosci.* 11, 127–138. doi: 10.1038/nrn2787

Friston, K. (2013). Life as we know it. *J. R. Soc. Interface* 10, 475. doi: 10.1098/rsif.2013.0475

Friston, K., Rigoli, F., Ognibene, D., Mathys, C., FitzGerald, T., and Pezzulo, G. (2015). Active inference and epistemic value. *Cogn, Neurosci.* 6:187–214. doi: 10.1080/17588928.2015.1020053

Friston, K. J., Daunizeau, J., Kilner, J., and Kiebel, S. J. (2010). Action and behavior: a free-energy formulation. *Biol, Cybern.* 102, 227–260. doi: 10.1007/s00422-010-0364-z

Friston, K. J., Shiner, T., FitzGerald, T., Galea, J. M., Adams, R., Brown, H., et al. (2012). Dopamine, affordance and active inference. *PLoS Comput, Biol.* 8, e1002327. doi: 10.1371/journal.pcbi.1002327

Gibson, J. J. (1986). *The Ecological Approach to Visual Perception, Vol. 1.* New York, NY: Psychology Press.

Gumbsch, C., Butz, M. V., and Martius, G. (2021). "Sparsely changing latent states for prediction and planning in partially observable domains," in *Advances in Neural Information Processing Systems*, Vol. 34, eds M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. W. Vaughan (Curran Associates), 17518–17531. Available online at: https://proceedings.neurips.cc/paper/2021/file/927b028cfa24b23a09ff20c1a7f9b398-Paper.pdf.

Ha, D., and Schmidhuber, J. (2018). World models. *arXiv preprint arXiv:1803.10122.* doi: 10.5281/zenodo.1207631

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. (2019a). Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603.* doi: 10.48550/arXiv.1912.01603

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., et al. (2019b). "Learning latent dynamics for planning from pixels," in Proceedings of the 36th International Conference on Machine Learning, Vol. 97, eds K. Chaudhuri and R. Salakhutdinov (PMLR), 2555–2565. Available online at: http://proceedings.mlr.press/v97/hafner19a/hafner19a.pdf

Lenz, I., Knepper, R., and Saxena, A. (2015). "Deepmpc: Learning deep latent features for model predictive control," in *Proceedings of Robotics: Science and Systems* (Rome).

Levine, S., Kumar, A., Tucker, G., and Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643.* doi: 10.48550/arXiv.2005.01643

Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., and Stoica, I. (2018). Tune: a research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118.* doi: 10.48550/arXiv.1807.05118

O'keefe, J., and Nadel, L. (1978). *The Hippocampus as a Cognitive Map.* Oxford: Clarendon Press.

Otte, S., Schmitt, T., Friston, K., and Butz, M. V. (2017). "Inferring adaptive goal-directed behavior within recurrent neural networks," in *Artificial Neural Networks and Machine Learning-ICANN 2017, volume 10613* (Cham: Springer International Publishing), 227–235.

Pfeiffer, B. E., and Foster, D. J. (2013). Hippocampal place-cell sequences depict future paths to remembered goals. *Nature* 497, 74–79. doi: 10.1038/nature12112

Pinneri, C., Sawant, S., Blaes, S., Achterhold, J., Stueckler, J., Rolinek, M., et al. (2020). Sample-efficient cross-entropy method for real-time planning. *arXiv preprint arXiv:2008.06389.* doi: 10.48550/arXiv.2008.06389

Qi, W., Mullapudi, R. T., Gupta, S., and Ramanan, D. (2020). Learning to move with affordance maps. *arXiv preprint arXiv:2001.02364, ICLR 2020.* doi: 10.48550/arXiv.2001.02364

Rubinstein, R. (1999). The cross-entropy method for combinatorial and continuous optimization. *Methodol. Comput. Appl. Probab.* 1, 127–190. doi: 10.1023/A:1010091220143

Sutton, R. S., and Barto, A. G. (2018). *Reinforcement Learning: An Introduction.* MIT Press.

Tani, J. (2017). Dialogue: exploring robotic minds by predictive coding principle. *IEEE CDS Newslett.* 14, 4–13.

Tolman, E. C. (1948). Cognitive maps in rats and men. *Psychol. Rev.* 55, 189. doi: 10.1037/h0061626

Zacks, J. M., Speer, N. K., Swallow, K. M., Braver, T. S., and Reynolds, J. R. (2007). Event perception: a mind-brain perspective. *Psychol. Bull.* 133, 273–293. doi: 10.1037/0033-2909.133.2.273

Zacks, J. M., and Tversky, B. (2001). Event structure in perception and conception. *Psychol. Bull.* 127, 3. doi: 10.1037/0033-2909.127.1.3