Multi-Modal Natural Intelligence through Active Predictive Coding

Jeffrey Duan*, Vishwas Sathish[1]*, Crimson Stambaugh[1], Rajesh P. N. Rao[1,2]

[1]Paul G. Allen School of Computer Science and Engineering
University of Washington, Seattle, WA 98195, USA
duanjeffrey@gmail.com, vsathish@cs.washington.edu, cstamb@uw.edu, rao@cs.washington.edu
[2]Corresponding author

Abstract

Active predictive coding (APC) is a recently proposed theory of the neocortex that postulates that a canonical sensorymotor processing circuit is replicated across cortical areas. These areas are organized in a rough hierarchy, with higher-level neural states modulating lower-level circuits implementing state-transition dynamics and policy functions. Such a structure enables the network to learn the compositional structure of the world, allowing it to rapidly compose solutions to new problems and generalize quickly to new environments. In APC, complex state transition dynamics are modeled as a sequence of simpler dynamics, which in turn are modeled using even simpler dynamics, and so on. Complex policies are similarly modeled as sequences of simpler policies, with the lowest level comprising sequences of primitive actions. Here we show that the APC model offers a unifying framework for multi-modal intelligence by demonstrating that the *same architecture* can (a) perform visual object recognition via active sensing (eye movements) and parts-based understanding, (b) navigate to desired goal locations in a complex environment through hierarchical planning, (c) learn to parse language hierarchically, infer the goal (i.e., intent) of an uttered sentence, and achieve the inferred goal through actions, and (d) scale up to realistic environments. Our results suggest that neurally-inspired approaches such as APC can help pave the way for more interpretable, generalizable, efficient, and human-like multi-modal AI.

Introduction

Predictive coding (Rao and Ballard 1999; Keller and Mrsic-Flogel 2018; Jiang and Rao 2022) and related theories such as active inference and the free energy principle (Friston and Kiebel 2009; Friston, Daunizeau, and Kiebel 2009) have garnered increasing attention in recent years as computational models of how the brain perceives and acts in the real world. In the traditional predictive coding model (Rao and Ballard 1999), different areas of the neocortex together implement a hierarchical generative model of the world. Feedback connections from a higher to a lower level predict lower-level responses. Prediction errors propagate via feedforward connections to update higher-level estimates during inference, and at a slower timescale, update network weights for learning, thereby offering a biologically plausible and efficient alternative to backpropagation.

While the original formulation of predictive coding ignored actions, the recent theory of active predictive coding (APC) (Rao, Gklezakos, and Sathish 2023; Rao 2024) acknowledges the central role of actions in cortical function and integrates actions and sensory states in hierarchical feedback loops. The resulting architecture serves two important functions: (a) the complex state-transition dynamics of real world environments can be modeled as sequences of simpler dynamics, which in turn can be decomposed into even simpler dynamics, and so on; (b) complex action sequences can similarly be decomposed into sequences of simpler action sequences or policies (state-to-action mappings), with the lowest level composed of sequences of primitive actions. It has been hypothesized (Rao 2024) that evolution may have arrived at such an architecture to enable the brain to learn and exploit the compositional nature of the world we live in, allowing us to rapidly compose solutions to new problems that may confront us and generalize quickly to new environments. The APC model has been shown to provide explanations for both the neuroanatomical structure of the neocortex (laminar organization and cortico-cortical feedback connections) as well as properties of neurons that have been observed across cortical areas (see (Rao 2024) for details).

In this paper, we show that the APC model provides a unifying framework for multi-modal intelligence. We present results demonstrating that the *same APC architecture* can (1) perform visual object recognition via active sensing (eye movements) and parts-based understanding, (2) navigate to desired goal locations in a large environment through hierarchical planning, (3) learn to parse language hierarchically, infer the goal (i.e., intent) of an uttered sentence, and achieve the inferred goal through actions, and (4) scale up to realistic environments.

Active Predictive Coding

The evolution of intelligence may have been predicated on the ability to move (Llinás 2001): early nervous systems likely developed intelligence to enable an organism to sense the environment and move towards nutrients required for survival and away from noxious stimuli and predators. Several hundred million years later, we still find a tight connec-

*These authors contributed equally.

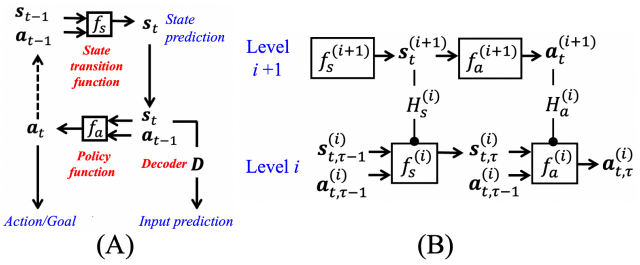


Figure 1: **Active Predictive Coding**. (A) Active predictive coding model for a single level. (B) Hierarchical active predictive coding model. See text for details.

tion between sensation and action maintained through evolution and in the brains of mammals like mice and humans. For example, recent large-scale recordings across the neocortex in mice have revealed (Zatka-Haas et al. 2021) that almost all areas, including areas traditionally labeled as "sensory cortex," are influenced by upcoming actions. Regions of the brain responsible for executing actions send an "efference copy" of an impending action to other brain regions such as the cortex to allow them to update their neural representations in anticipation of expected sensory stimuli.

Active predictive coding (APC) (Rao 2024; Rao, Gklezakos, and Sathish 2023) is a recent extension of predictive coding that acknowledges the central role of actions (e.g., eye, head, body movements) in perception, cognition and behavior. The model is motivated by the fact that each cortical area, even primary sensory areas such as V1 (primary visual cortex), not only receives sensory information but also sends outputs to the brain's evolutionarily-older motor centers such as the superior colliculus and the spinal cord.

APC State-Action Networks

The APC model assumes that each cortical area includes a "state-prediction" (or state-transition) function f_s that predicts the next sensory state s_t at time t given a previous state s_{t-1} and action a_{t-1} (see Figure 1A, top). The same cortical area also computes an action or "policy" function f_a that maps the current state s_t (and previous action a_{t-1}) to the next action a_t that is appropriate for the current task or goal (Figure 1A, bottom).

The recurrent neuronal networks implementing f_s and f_a feed their outputs to each other as shown in Figure 1A, defining a generative model and leading naturally to a sequence of predictions of states and actions. Such a sequence can track environmental states during behavior or alternatively, be used for internal simulation and planning. During behavior, an efference copy of an action selected for execution is sent to the state-transition function for generating the next state prediction (dashed arrow in Figure 1A), while sensory input prediction errors (after an action is executed) are conveyed back to the network to update the state s_t as part of the inference process. The same prediction errors are also used to update the parameters of f_s and learn the state transition dynamics of the environment (see (Rao, Gklezakos, and Sathish 2023) for details). The policy function f_a can be learned through model-based planning and reinforcement learning (see examples below).

Hierarchical APC Model

An important feature of the human nervous system is its hierarchical organization in both sensory and motor domains. Evolution builds new capabilities on top of what is already available, and solutions to new problems in new ecological niches could be potentially arrived at by adding a higher level network that combines previously evolved lower-level solutions (Kaas 2008; Mengistu et al. 2016). Solving a complex problem by breaking it down into simpler sub-problems is also a computationally sensible strategy: for example, consider the problem of going to the grocery store. At a high level of abstraction, you may divide the task into subtasks (or "sub-goals") such as walking to the door of your house from whichever room you are currently in, opening the door, walking to where your car is parked, getting into car, etc. Note that many of these sub-tasks can be *re-used* for solving a range of other problems (e.g., going to work, going to visit a friend, etc.). Therefore, it may be useful to learn a policy for the sub-task to avoid planning actions each time the sub-task is re-used as part of a different task; indeed, this is precisely the motivation behind well-known hierarchical reinforcement learning (HRL) frameworks such as options (Sutton, Precup, and Singh 1999; Barto and Mahadevan 2003a; Abel 2022). The ability to divide a problem into components and re-use the components to solve new problems is at the heart of *compositionality*, a powerful attribute underlying the flexibility and fast generalization ability of human cognition (Lake et al. 2017; Smolensky et al. 2022; Ellis et al. 2021; Andreas 2019).

Note that compositionality for an agent interacting with the physical world can involve not only dividing a complex action into simpler actions but also dividing a complex state, defined by its state transition function, into a sequence of simpler state transition functions. For example, an environment (e.g., an office building or gridworld maze) can be divided into its compositional elements such as rooms and corridors. Simiarly, a visual object can be decomposed into a sequence of parts and where each part appears within the object's reference frame.

The APC model can be made hierarchical as follows (Figure 1B): a state representation *vector* $s^{(i+1)}$ at abstraction level $i+1$ generates or modulates (via a hypernet H_s^i (Ha, Dai, and Le 2017; Rao 2024)) a state transition *function* $f_s^{(i)}$ at the lower level i (along with an initial state vector $s_0^{(i)}$ to start the lower-level state-action sequence); similarly, a higher-level action *vector* $a^{(i+1)}$ generates (via a hypernet H_a^i) a lower-level policy *function* $f_a^{(i)}$ ("option" in RL (Sutton, Precup, and Singh 1999; Abel 2022)). The two lower-level functions interact with each other, as in Figure 1A, to generate lower-level states and actions as shown in Figure 1B. Each such state vector and action vector can in turn generate transition and policy functions at an even lower level of abstraction. A lower-level sequence executes for a period of time until a condition is met (e.g., a sub-goal is reached, a task is completed or times out, or there is an ir-

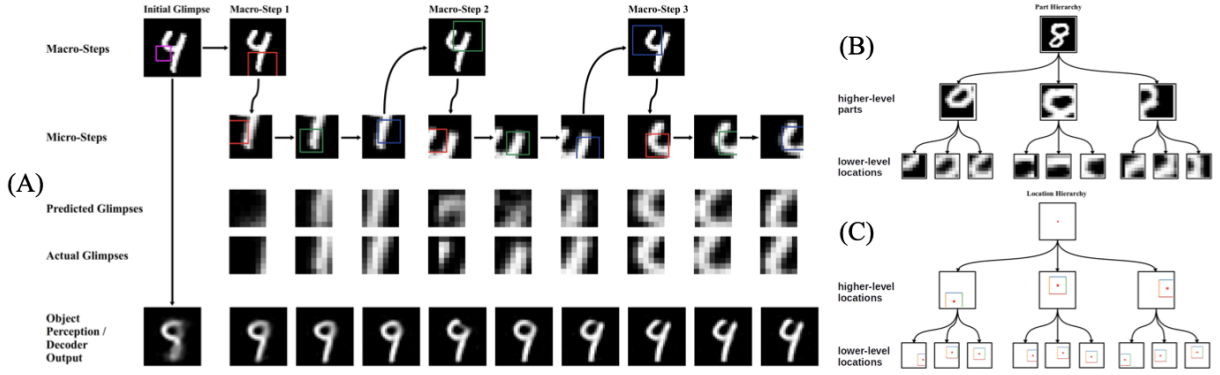


Figure 2: **Active Vision**. **(A)** 2-Level hierarchical inference of an object (MNIST digit) via glimpses obtained through active sensing using eye movements. **(B)** Example of a learned part-whole hierarchy for the digit "8" as represented by two levels of parts and their locations with the level's reference frame. Adapted from (Rao, Gklezakos, and Sathish 2023).

reconcilable error at that level). Then, control returns to the higher-level, which transitions to a new higher-level state and a new higher-level action, and the process continues. It is clear that such a generative model can generate the dynamics of the states (the "physics" of the world) and action sequences (produced by a policy) at different time scales, providing a mathematical and hierarchical way of describing complex tasks (such as going to the grocery store).

Active Vision

We first illustrate the applicability of APC in combining the modalities of vision and action. Given a task, humans employ eye movements to move the high-resolution fovea to appropriate locations in a scene, gathering information useful for solving the task (Yarbus 1967). The APC architecture can implement such an "active vision" approach to visual perception by virtue of its integrated state-action networks.

To illustrate this capability, we simulated a 2-level APC model (Figure 1B) (Rao, Gklezakos, and Sathish 2023) in which the lower-level actions emulated eye movements (or "attention") by moving a fovea ("glimpse sensor" (Mnih et al. 2014)) to extract high-resolution information about a small part of a larger input image. At each higher-level time step t (a "macro-step"), the higher-level action vector $a_t^{(2)}$ generates two values: a 2D location vector L_t and a "macro-action" (or option) vector o_t. The location L_t is used to restrict the bottom level to a sub-region $I_t^{(1)} = G(I, L_t, M)$ of scale M centered around L_t, corresponding to a new frame of reference selected by the top level within the input image. The option vector o_t, which operates over this frame of reference, is used as an input to the top-down network $H_a^{(1)}$, which generates (or modulates) the lower-level action network ($f_a^{(1)}$ in (Fig. 1B).

The lower-level action network in turn generates a new action $a_{t,\tau}^{(1)}$ at lower level time step τ (a "micro-step"), which selects a fixation location within the larger reference frame of $I_t^{(1)}$ specified by the higher level. This fixation yields a small high-resolution "glimpse" (foveal image), specifying a nested reference frame within the larger one. The lower-level action also predicts a new state vector $s_{t,\tau+1}^{(1)}$ which generates, via a trained decoder, a prediction for the glimpse image expected after the "eye movement." The resulting prediction error is used to update the state vector, as prescribed by the predictive coding model (Rao and Ballard 1999). For the results below, the state networks at both levels were trained to minimize image prediction errors while the action networks were trained using reinforcement learning for the task of image reconstruction (alternately, image classification could be specified as the task; see (Rao, Gklezakos, and Sathish 2023) for details)). The APC model was tested on the MNIST dataset (10 classes of handwritten digits), Fashion-MNIST (10 classes of clothing items) and Omniglot (1623 hand-written characters from 50 alphabets).

Figure 2A shows an example of a learned parsing strategy by the 2-level APC model for a handwritten digit. The higher level learned to select actions that cover the input image sufficiently, avoiding blank regions, while the lower level learned to parse sub-parts inside the reference frame computed by the higher level. Note how the initially uncertain hypothesis (blurry "9") is refined as the model makes "eye movements" to accumulate evidence, converging on the identity of the input digit ("4"). Figure 2A also suggests why our perception appears stable despite dramatic changes in our retinal images as our eyes move to sample a scene: the last row of the figure shows how the recurrent network in the model maintains a visual hypothesis that is gradually refined, not exhibiting the large changes seen in the "Actual Glimpses" row of Figure 2A. This "perceptual stability" is enabled by the model's ability to predict the expected glimpses for planned "eye movements" (Figure 2A, Predicted Glimpses), similar to predictive activity observed in the visual cortex before eye movements (Nakamura and Colby 2002; Duhamel, Colby, and Goldberg 1992).

Figure 2 also shows a learned part-whole hierarchy for an MNIST input, in the form of a parse tree of parts and subparts (strokes and mini-strokes) (Fig. 2B) along with their locations within nested reference frames (Fig. 2C). The model learns different parsing strategies for different classes of objects. In other experiments, we investigated the predictive

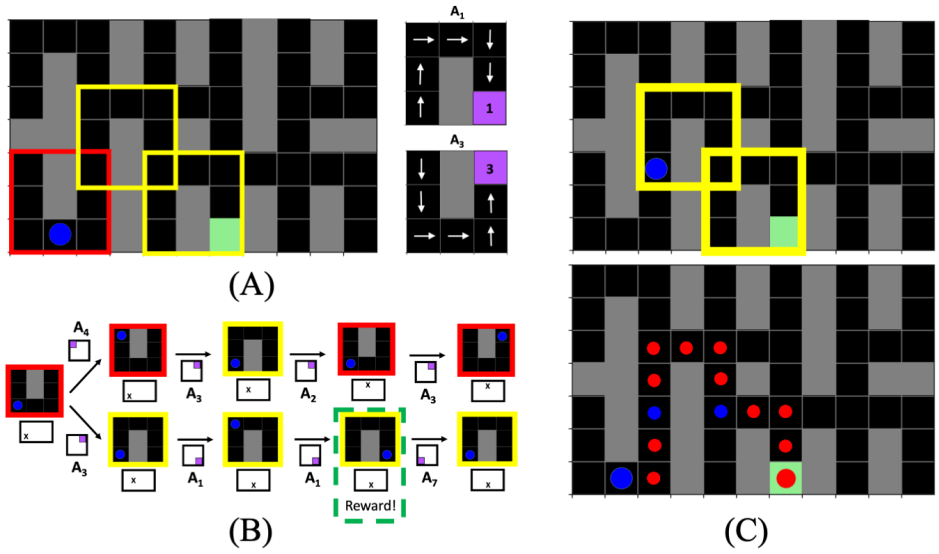


Figure 3: **Hierarchical Planning for Navigation**. (A) The problem of navigating in a large environment (left panel) can be reduced to planning using high-level states (red and yellow outlined "rooms") and high-level abstract actions (panels on the right show two abstract actions: A_1 and A_3). Blue: current location, gray: walls, green: current goal location. (B) To navigate to the goal, the APC model uses its learned high-level state network to sample K high-level state-action sequences ($K = 2$ here, shown bifurcating from the initial state). In each sequence, the high-level state is depicted by a predicted room image (red or yellow outlined image) and its location (marked by an "X" in the rectangular global frame below the image). High-level actions: square local frames (below arrows) with goal locations (purple). (C) Given sampled sequences, the model picks the sequence with highest total reward, executes this sequence's first (high-level) action to reach the blue location (top panel), and repeats to reach the goal location with only 3 high-level actions (bottom panel). Small red dot: intermediate location; small blue dot: intermediate goal. Adapted from (Rao, Gklezakos, and Sathish 2023).

and generative ability of the trained model by setting the prediction error input to the lower level network to zero, which disconnects the model from the input and forces it to predict a sequence of parts to "complete" an object. We found that the model generates plausible predictions of parts given an initial glimpse (Rao, Gklezakos, and Sathish 2023).

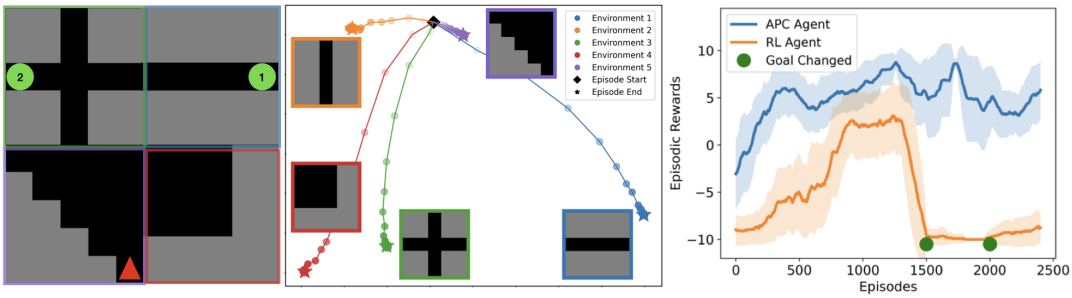
Navigation and Hierarchical Planning

The same architecture used above for active vision can also be used for planning hierarchical actions for navigation. Consider the problem of navigating from any starting location to any goal location in a large "multi-rooms" building environment such as the one in Figure 3A (gray: walls, blue circle: current location, green square: current goal location). Adopting the convention of reinforcement learning (RL) (Sutton and Barto 2018), we regard the lower-level states of the APC model in this case as discrete locations in the grid, and lower-level actions as going north (N), east (E), south (S) or west (W). A large reward (+10) is received at the goal location, with a small negative reward (-0.1) for each action to encourage short paths.

Just as an object (e.g., the MNIST digit in Fig. 2B) is modeled in APC as consisting of parts (e.g., strokes, curves) at different locations, an environment such as the one in Figure 3A is modeled as composed of smaller components (here, two 3×3 "room types" S1 and S2 which act as local reference frames), outlined in red and yellow respectively in the figure, occurring at different locations in the global reference frame. The higher-level states of the APC model are consequently defined by state embedding vectors S1 and S2, trained to generate, via the top-down network $H_s^{(1)}$ (Fig. 1B), the lower-level transition functions $f_s^{(1)}$ for rooms S1 and S2 respectively.

Similar to how the APC vision model reconstructs an image by composing parts from sub-parts (Fig. 2), the APC model for planning computes higher-level action embedding vectors $\mathbf{A}_i$ (option vectors) for the multi-rooms world that generate, via top-down network $H_a^{(1)}$ (Fig. 1B), lower-level policies $f_a^{(1)}$ that produce primitive actions (go 1 step in direction N, E, S, or W) from any location in the local reference frame (S1 or S2) to reach the goal i within that frame. For the present simulation, eight macro-actions (options) $\mathbf{A}_i$ were trained using reinforcement learning to reach one of the four corners of S1 or S2 (see (Rao, Gklezakos, and Sathish 2023) for details). Figure 3A (right panels) illustrate two of the eight bottom-level policies generated by higher-level action vectors $\mathbf{A}_1$ and $\mathbf{A}_3$ for reaching goal locations 1 and 3 respectively. Defining these policies to operate within the local reference frame of the higher-level state S1 or S2 (regardless of global location in the building) confers the APC model with great flexibility because the same policy can be *re-used at multiple locations* to solve local tasks (here, reach sub-goals within room types S1 or S2).

The higher-level state network was trained to predict the next higher-level state (decoded as an image of room type S1 or S2, plus its location) given the current higher-level state and higher-level action. This trained higher-level state net-



(A) (B) (C)

Figure 4: **Higher-Level State Inference for Navigation and Robustness to Goal Changes. (A)** An environment composed of four types of rooms, each defined by a different transition function $f_s^{(1)}$ characterizing the structure of the walls in the room. **(B)** Examples of inference of higher-level state as the agent explores and interacts with the walls of a room: each colored trajectory shows successive updates to the state vector for a particular room based on backpropagating the input prediction error at each time step through the state hypernetwork which generates the transition function. **(C)** Performance of an APC agent using hierarchical planning for navigation compared to a traditional RL agent in a goal-changing multi-rooms environment. Plot (C) adapted from (Rao, Gklezakos, and Sathish 2023).

work was used for planning a sequence of 4 higher-level actions (using model predictive control (MPC) (Richards 2004)) as follows: random state-action trajectories of length 4 were generated using the higher-level state network by starting from the current state and picking one of the four random actions A_i for each next state; the action sequence with the highest total reward was selected and its first action was executed, and this process was repeated. Figures 3B-C illustrate this planning process using the trained APC model.

Figures 4A and 4B provide examples of how the APC agent can infer what type of room it is in by inferring the higher-level state that generates the corresponding transition function $f_s^{(1)}$, via exploratory movements within the room – the evolution of the higher-level state vectors and the final converged vectors for the four rooms in Figure 4A are shown as colored trajectories ending in a star in Figure 4B (visualized as a 2D PCA projection of the state vectors). Figure 4C illustrates how the APC model can re-use its learned high-level actions in new combinations to plan hierarchically and quickly solve novel tasks such as when the goal location is changed, whereas a traditional reinforcement learning agent is unable to cope with the goal change and must learn a new policy from scratch to navigate to the new goal location.

Language Understanding and Multi-Modal Problem Solving

To further illustrate the applicability of APC for achieving multi-modal intelligence, we demonstrate how the APC model used above for vision and navigation can also parse natural language and ground its understanding of linguistic terms in sensed objects and actions. Specifically, we show how a 2-level APC model can both (a) parse a natural language statement specifying a goal and (b) fulfill that goal by perceiving relevant objects within its environment and executing actions.

BabyAI is a research platform designed to study grounded language learning (Chevalier-Boisvert et al. 2018). It includes a collection of 2D gridworld environments with missions specified by natural language sentences. The goal of the agent is to understand and fulfill those missions by navigating in the gridworld environment with partial observability and interacting with objects. The object types include balls, boxes, keys, and doors which can be one of 6 colors (red, yellow, green, blue, purple, grey). The natural language sentences are generated by a context-free grammar.

Here, we focused on one-room environments with the following missions: "go to a/the {color} {object type}," "pick up a/the {color} {object type}," and "open a/the {color} door." To demonstrate language understanding, we also included mission types "put a/the {color} {object type} next to a/the {color} {object type}." The above mission statements have verbs (go, open, pick up, put), adjectives (colors), and nouns (objects) that the agent must learn to understand and map to its own actions and perceptions. To achieve these missions, the agent can perform 7 different actions—left, right, forward, pickup, drop, toggle (interact with doors), and done (signal task completion). These actions allow the agent to navigate and interact with objects in the gridworld environment. The agent must learn through training to associate different parts of the mission sentence to relevant features and objects in the world, and associate the verbs with appropriate actions it needs to take. In this way, the agent "grounds" its understanding of natural language in its own sensory perceptions and motor actions.

For both language understanding and executing actions to achieve a goal, we use a 2-level APC model employing state and action hypernetworks $H_s^{(1)}$ and $H_a^{(1)}$ respectively (Fig. 1B). Rather than generating an entire network at the lower level, we adopt an embedding approach (Galanti and Wolf 2020), where the hypernetwork produces an embedding vector from the higher-level latent vector, which is then used to modulate a lower-level recurrent network. The lower-level network (we used an LSTM network (Hochreiter and Schmidhuber 1997) for language and a recurrent neural network (RNN) for action) takes as input this top-down embedding, concatenates it to the previous state $s_{t,\tau-1}^{(1)}$ and action $a_{t,\tau-1}^{(1)}$, and predicts the next state or action (Fig. 1B).

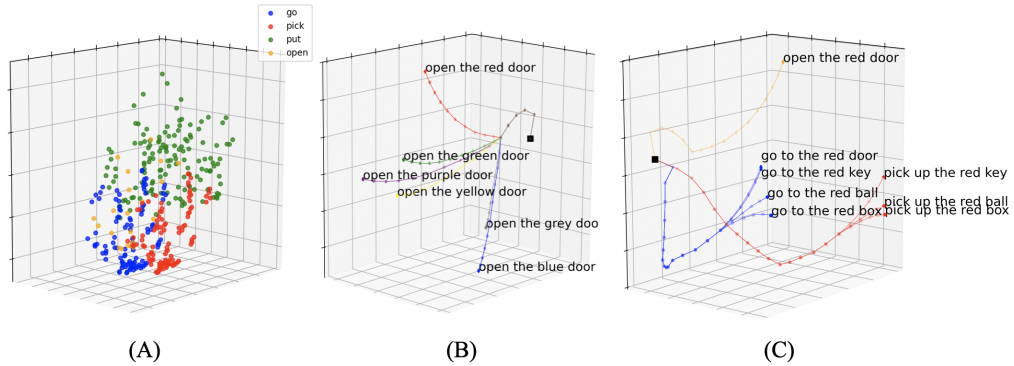


Figure 5: **Language Understanding: Parsing sentences and inferring goals/intent using APC. (A)** 3D PCA projection of inferred higher-level state for different mission sentences in BabyAI, colored based on the first word in the sentence. **(B)** Examples of higher-level state inference as the agent iterates over sentences beginning with the verb "open," one word at a time. Each colored trajectory shows successive updates to the higher-level state vector in a 3D PCA projected space based on backpropagating the prediction error at each time step through the language state hypernetwork that generates the lower-level transition function. **(C)** Examples of inference trajectories for different sentences containing the adjective "red." Note the convergence to different regions with local differentiation within each region.

As in the Active Vision and Navigation applications above, the higher-level state vector (in this case, the goal/intent of a sentence) is again inferred via backpropagation of prediction errors during inference, given a sequence of words. We first vectorize the words in the sentence by mapping them to a unique 34-dimensional one-hot vector (vocabulary size for possible BabyAI missions is 34). The "semantic meaning" of a sentence is captured by a 16-dimensional higher-level state vector $s^{(2)}$ (Fig. 1B): this state vector is fed as input to the language hypernetwork $H_s^{(1)}$ in Fig. 1B to generate a 16-dimensional embedding vector $\mathbf{e}_l$ as input to modulate the lower-level LSTM network $f_s^{(1)}$ predicting words.

The inference process makes updates to the higher-level state $s^{(2)}$ (within the "semantic space" of possible sentences) as evidence accumulates word by word. Inference of $s^{(2)}$ minimizes prediction loss (negative log likelihood loss) using gradient updates for each lower-level time step τ. We updated $s^{(2)}$ 5 times per word as we sequentially process each word in the sentence. The hypernetwork and lower-level LSTM parameters are frozen during inference. During training, we perform inference to get the model's current best guess for $s^{(2)}$, from which we generate the embedding vector $\mathbf{e}_l$ via the hypernetwork and obtain the predicted sentence via the lower level LSTM. The predicted sentence is compared to the actual sentence and the error backpropagated to update the hypernetwork and LSTM parameters.

Figure 5A shows a 3D PCA projection of converged higher-level states $s^{(2)}$ for all the sentences the 2-level APC state network was trained on – note the clustering of sentences based on the verb type in different regions of the space. Figures 5B and 5C depict 3D PCA plots of $s^{(2)}$ converging for sentences beginning with the verb "open" (for different doors) and a comparison with two other verbs, respectively. We see striking parallels here between these word-by-word semantic inference trajectories for language, the glimpse-by-glimpse object inference process for active vision (Figure 2A) and the move-by-move room recognition trajectories for navigation (Figure 4B), indicating the generality of the APC framework. In language understanding, object recognition, and navigation, the higher-level state represents a hypothesis that is iteratively refined using prediction errors to arrive at a stable, higher-level state that captures the intent behind the sentence, identity of the object, and identity of the room, respectively.

We ground the semantic understanding of the linguistic mission statement in perception and action by implementing a 2-level APC action network for hierarchical actions in the BabyAI gridworld. At each macro-step t, we have a higher-level policy $f_a^{(2)}$ ($f_a^{(i+1)}$ in Fig. 1B) that takes the current gridworld input and the higher-level state $s_t^{(2)}$ inferred from language, and outputs a higher-level action $a_t^{(2)}$ (one of pickup, drop, toggle, done, or "go to the {color} {object type}"). This higher-level policy was learned using an actor-critic model, with an LSTM and an MLP feature extraction layer trained using the PPO reinforcement learning method (Schulman et al. 2017). The "go to" higher-level actions were vectorized as one-hot vectors and used as input for the action hypernetwork $H_a^{(1)}$ (similar to the language hypernetwork $H_s^{(1)}$). The action hypernetwork generates a 64-dimensional embedding vector $e_a = H_a^{(1)}(a_t^{(2)})$ for the lower-level RNN implementing $f_a^{(1)}$: $h_\tau = \tanh(x_\tau W_1 + b_1 + h_{\tau-1} W_2 + b_2)$ and $\hat{a}_{\tau+1} = ReLU(h_\tau W_3 + b_3)$, where $x_\tau = [e_a, s_\tau^{(1)}, a_\tau^{(1)}]$ is the concatenated lower-level input and $[\theta, W_{1:3}, b_{1:3}]$ are the model parameters. Here, $s_\tau^{(1)}$ is the agent's lower-level state obtained by concatenating the vectors representing the agent's orientation and relative position from the starting point, and a 7×7 overhead image of the gridworld in front of the agent. The action hypernetwork $H_a^{(1)}$ was trained using the DAgger imitation learning algorithm (Ross, Gordon, and Bagnell 2011), with the expert trajectories generated by the BabyAI-provided expert bot.

We trained the higher-level policy $f_a^{(2)}$ on four BabyAI

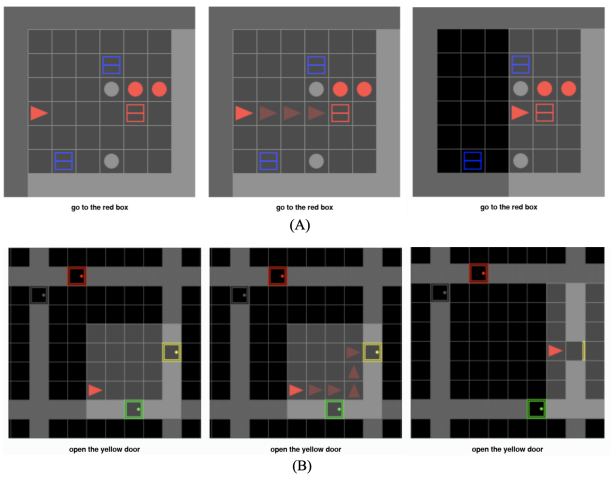


Figure 6: **Achieving Goals Inferred from Understanding Language**. **(A)** The agent achieving the goal "go to the red box" in the BabyAI environment BabyAI-GoToLocalS8N7-v0.The higher-level actions (subgoals) the higher-level policy outputs are "go to the red box" (executes the corresponding learned lower-level policy) and "done." **(B)** The agent achieving the goal "open the yellow door" from the BabyAI environment BabyAI-OpenDoorColor-v0. The higher-level actions that the higher-level policy outputs are "go to the yellow door" (executes the corresponding learned lower-level policy), "toggle" (opens the door), and "done."

environments: BabyAI-GoToObj-v0 (a one-room environment with only one object, where the mission is to go to that object), BabyAI-GoToLocalS8N7-v0 (a one-room environment with many objects, most of which are distractors), BabyAI-PickupDistDebug-v0 (similar to GoToLocal, a one-room environment with many distracting objects), and BabyAI-OpenDoorColor-v0 (a one-room environment with no objects, only doors in the walls surrounding the room). We achieved success rates of 99.40%, 75.30%, 43.40%, and 95.70% respectively for these BabyAI environments, where success is defined as completing the mission stated by the natural language sentence, and outputting the "done" subgoal within 100 macrosteps. Figure 6A and B show examples of successful mission completion trajectories generated by our trained 2-level APC model.

Scaling APC: Multi-Modal Navigation in the Habitat Environment

We tested whether APC scales to more realistic problems using the Habitat 2.0 environment (Szot et al. 2021), chosen for its efficient egocentric rendering, configuration flexibility, and a realistic suite of embodied AI tasks based on noisy and sometimes partially occluded observations. A sample environment and egocentric view of the agent is shown in Figure 7A.

The goal in Habitat is similar to the one in the *Navigation and Hierarchical Planning* section. An agent begins at a random location in a house and must navigate to a particular location specified by a position, GPS coordinates, or a language command (name of a target object or room) (Szot

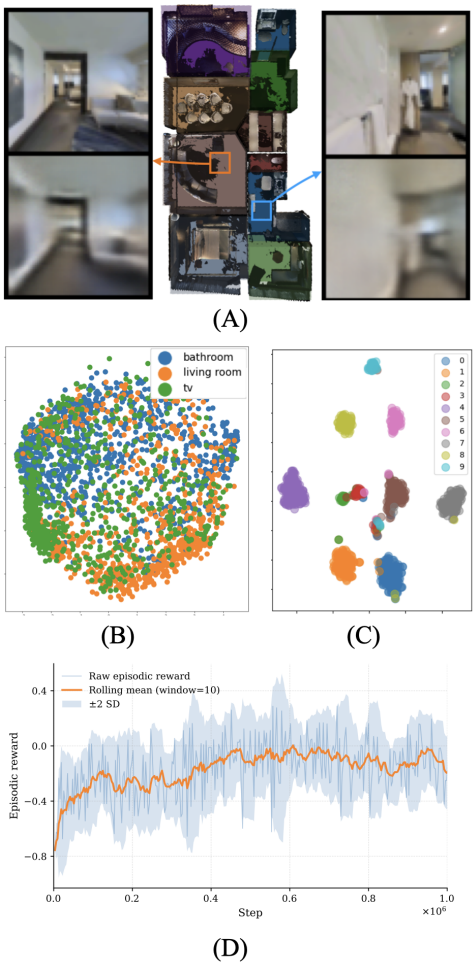


Figure 7: **Multi-Modal Navigation with APC in Habitat**: **(A)** We train a room-aware world model to predict next frames in an egocentric navigation environment. Our APC agent predicts the future frame while maintaining a higher-level state that estimates the current room. Top frame: the ground truth frame; Bottom frame: model's prediction. **(B)** 2D PCA projection of inferred higher-level states colored by the agent's true current room. **(C)** 2D PCA projection for the same higher-level states but now for all rooms in the environment and with an additional contrastive learning objective to improve clustering (see text). The resulting separation of inferred higher-level states allows us to train a 2-level APC model that navigates between rooms. **(D)** Performance of an action network trained using deep Q-learning to reach a target room from any location in the environment.

et al. 2021). When the agent reaches the goal location, the episode is considered successful. Here, we consider the "Go to" task of navigating to a particular room specified by a language command ("kitchen," "living room," etc.) from any starting location. The agent has three discrete actions (forward, turn left by 30 degrees, turn right by 30 degrees). Only the current egocentric image (no GPS, compass or additional sensory information) is provided to the agent during navigation. In reinforcement learning, this task is framed as learning a goal-conditioned policy (Liu, Zhu, and Zhang 2022).

Solving such navigation tasks can be difficult as a result of (1) the large state space due to pixel-based image inputs and (2) sparse reward structure (reward obtained only after reaching the goal). Several existing methods fail to effectively learn navigation in such scenarios. For example, (Wijmans et al. 2020) report that the PPO method takes up to 2.5 billion time steps to converge to a success rate of 15% without the use of GPS and compass sensors to guide the agent. However, even without GPS sensors, there is enough information in the sequence of image-based states to map sections of the environment. While traditional RL methods fail to exploit such structural information from the environment, we found that APC can map the entire environment with state-action hierarchies while navigating successfully.

We begin with the same 2-level APC architecture used in previous sections. Similar to the action embedding input in the previous section, the embedding input for the lower-level state network is computed as $e_s = H^{(1)}(s^{(2)})$. We encode images I_t with a CNN visual encoder to learn a representation of the image. The concatenated image representation, embedding vector from the higher state, and the previous action comprise the lower-level state network input vector $x_t = [Enc_\theta(I_t), e_s, a_t]$. The higher-level state $s^{(2)}$ is first inferred, allowing e_s to be computed. The lower-level state network then predicts the next state which is decoded to predict the next input $\hat{I}_{t+1} = Dec_\theta(x_{t+1})$. Examples of predicted images from the trained model are shown in Figure 7A. The inferred higher states $s^{(2)}$ for a sequence of steps in three different rooms of an environment are shown in Figure 7B. We use a dimension of 32 for higher-level states $s^{(2)}$ in all our Habitat experiments.

Unlike the gridworld examples in Figure 4B, the inferred higher states for different Habitat rooms do not cleanly separate into different regions of the 2D PCA space in Figure 7B, despite each room having unique structures and objects. This is due to the noisy nature of egocentric pixel-based inputs. For example, an egocentric image while the agent is in the kitchen can contain a view of the living room. Despite this noisy nature of egocentric views, we observe some clustering for different rooms (Figure 7B). However, this may not be practical for learning higher-level state-action networks for hierarchical planning or learning hierarchical policies (Barto and Mahadevan 2003b; Abel 2022; Rao, Gklezakos, and Sathish 2023). To overcome this issue, we introduce a contrastive learning objective (Khosla et al. 2021; Chen et al. 2020) to APC's reconstruction loss. For a given batch of episodes, the loss function is given by $\mathcal{L} = \mathcal{L}_{\text{reconstruct}} + \mathcal{L}_{\text{contrastive}}$ where $\mathcal{L}_{\text{reconstruct}} = \sum_{t=0}^{T} ||\hat{s}_{t+1}^{(1)} - s_{t+1}^{(1)}||_2^2$ is the standard predictive coding loss (also used in VAE and world model learning algorithms (Kingma and Welling 2013; Hafner et al. 2020, 2022)) and

$$\mathcal{L}_{\text{contrastive}} = \sum_{p \in P(i)} \log \frac{\exp(\mathbf{s}_i^{(2)} \cdot \mathbf{s}_p^{(2)}/\tau)}{\sum_{a \in A(i)} \exp(\mathbf{s}_i^{(2)} \cdot \mathbf{s}_a^{(2)}/\tau)} \quad (1)$$

where $P(i)$ is the set of positive examples for anchor i (samples from same room in Habitat, for example) and $A(i)$ is the set of all samples excluding the i^{th} anchor (different rooms). s_i is the representation of the i^{th} example and τ is a scalar temperature parameter (Khosla et al. 2021). For Habitat 2.0, we start each batch of episodes from different rooms and use the first few inferred $s^{(2)}$ as candidates to compute the contrastive loss. Representations from the same episode are considered as positive examples, and those from different episodes become the set of negative samples. This additional constraint helps the APC agent learn more discriminative higher states as seen in Figure 7C.

Similar to previous sections, we consider $a_r^{(2)}$ to be a higher-level action, in this case representing a goal-conditioned policy to reach room r. Given an inferred higher-level state $s^{(2)}$, inferred, for example, from language (see previous section), we can learn a mapping from a language command ("Go to the kitchen") to a higher-level action. As in previous sections, we learn the hypernetwork $H_a^{(1)}$ that generates for each higher-level action a lower level policy $f_a^{(1)}$. We use goal-conditioned Deep Q-Learning to train this hierarchical policy to navigate to all rooms (Mnih et al. 2013; Liu, Zhu, and Zhang 2022). A sparse reward of 1 was given when the agent successfully navigated to the center of the goal room and a penalty of -0.01 was given for every step taken. Preliminary results show that such a method allows for superior sample efficiency in navigating to a desired room (Figure 7D). Our ongoing work is focused on learning the higher-level state transition function $f_s^{(2)}$ for comparing our hierarchical planning-based approach to Deep Q-Learning based off-policy algorithms. We are also exploring the use of diverse language commands for Habitat ("Go To", "Pick Up", "Drop", etc), building on the BabyAI experiments described in the previous section.

Conclusion

Our results show how APC's hierarchical state-action architecture can provide a unifying framework for multi-modal intelligence. We showed how the same architecture can be used for solving problems in active vision, planning, navigation, language understanding and goal inference, concluding with examples illustrating multi-modal problem solving in the BabyAI and Habitat environments. The APC approach is inspired by the organizational principles of the neocortex, such as its laminar and hierarchical structure as well as the observation that cortical structure is remarkably similar across cortical areas. It has been shown that an area associated with one modality (e.g., audition) can learn to process signals from a different modality (e.g., vision) (Roe et al. 1992; von Melchner, Pallas, and Sur 2000), pointing to a common underlying computation that generalizes across modalities.

Our ongoing work is focused on scaling the APC model to larger-scale environments and RL benchmarks, and leveraging the model's compositional structure and ability to generate new transition functions on the fly to achieve fast transfer across environments and multi-modal tasks.

Acknowledgments This work was supported by AFOSR award no. FA9550-24-1-0313, NSF EFRI grant no. 2223495, and a TWCF "Frameworks" grant.

References

Abel, D. 2022. A Theory of Abstraction in Reinforcement Learning. arXiv:2203.00397.

Andreas, J. 2019. Measuring compositionality in representation learning. *arXiv preprint arXiv:1902.07181*.

Barto, A. G.; and Mahadevan, S. 2003a. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13: 341–379.

Barto, A. G.; and Mahadevan, S. 2003b. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13: 2003.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A Simple Framework for Contrastive Learning of Visual Representations. arXiv:2002.05709.

Chevalier-Boisvert, M.; Bahdanau, D.; Lahlou, S.; Willems, L.; Saharia, C.; Nguyen, T. H.; and Bengio, Y. 2018. BabyAI: First Steps Towards Grounded Language Learning With a Human In the Loop. *CoRR*, abs/1810.08272.

Duhamel, J. R.; Colby, C. L.; and Goldberg, M. E. 1992. The updating of the representation of visual space in parietal cortex by intended eye movements. *Science*, 255: 90–92.

Ellis, K.; Wong, C.; Nye, M.; Sablé-Meyer, M.; Morales, L.; Hewitt, L.; Cary, L.; Solar-Lezama, A.; and Tenenbaum, J. B. 2021. Dreamcoder: Bootstrapping inductive program synthesis with wake-sleep library learning. In *Proceedings of the 42nd acm sigplan international conference on programming language design and implementation*, 835–850.

Friston, K.; and Kiebel, S. 2009. Predictive coding under the free-energy principle. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1521): 1211–1221.

Friston, K. J.; Daunizeau, J.; and Kiebel, S. J. 2009. Reinforcement learning or active inference? *PloS one*, 4(7): e6421.

Galanti, T.; and Wolf, L. 2020. On the Modularity of Hypernetworks. In *Advances in Neural Information Processing Systems*, volume 33, 10409–10419. Curran Associates, Inc.

Ha, D.; Dai, A. M.; and Le, Q. V. 2017. HyperNetworks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Hafner, D.; Lee, K.-H.; Fischer, I.; and Abbeel, P. 2022. Deep hierarchical planning from pixels. *Advances in Neural Information Processing Systems*, 35: 26091–26104.

Hafner, D.; Lillicrap, T.; Norouzi, M.; and Ba, J. 2020. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation*, 9(8): 1735–1780.

Jiang, L. P.; and Rao, R. P. 2022. Predictive Coding Theories of Cortical Function.

Kaas, J. H. 2008. The evolution of the complex sensory and motor systems of the human brain. *Brain research bulletin*, 75(2-4): 384–390.

Keller, G. B.; and Mrsic-Flogel, T. D. 2018. Predictive Processing: A Canonical Cortical Computation. *Neuron*, 100: 424–435.

Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; and Krishnan, D. 2021. Supervised Contrastive Learning. arXiv:2004.11362.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Lake, B. M.; Ullman, T. D.; Tenenbaum, J. B.; and Gershman, S. J. 2017. Building machines that learn and think like people. *Behavioral and brain sciences*, 40: e253.

Liu, M.; Zhu, M.; and Zhang, W. 2022. Goal-Conditioned Reinforcement Learning: Problems and Solutions. arXiv:2201.08299.

Llinás, R. R. 2001. *I of the vortex: From neurons to self*. The MIT Press.

Mengistu, H.; Huizinga, J.; Mouret, J.-B.; and Clune, J. 2016. The evolutionary origins of hierarchy. *PLoS computational biology*, 12(6): e1004829.

Mnih, V.; Heess, N.; Graves, A.; and Kavukcuoglu, K. 2014. Recurrent Models of Visual Attention. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. A. 2013. Playing Atari with Deep Reinforcement Learning. *CoRR*, abs/1312.5602.

Nakamura, K.; and Colby, C. L. 2002. Updating of the visual representation in monkey striate and extrastriate cortex during saccades. *Proc Natl Acad Sci U S A*, 99: 4026–4031.

Rao, R. P.; Gklezakos, D. C.; and Sathish, V. 2023. Active Predictive Coding: A Unifying Neural Model for Active Perception, Compositional Learning, and Hierarchical Planning. *Neural Computation*, 36(1): 1–32.

Rao, R. P. N. 2024. A sensory-motor theory of the neocortex. *Nat Neurosci*, 27(7): 1221–1235.

Rao, R. P. N.; and Ballard, D. H. 1999. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2: 79–87.

Richards, A. 2004. *Robust constrained model predictive control.* Ph.D. thesis, MIT.

Roe, A. W.; Pallas, S. L.; Kwon, Y. H.; and Sur, M. 1992. Visual projections routed to the auditory pathway in ferrets: receptive fields of visual neurons in primary auditory cortex. *J Neurosci*, 12: 3651–3664.

Ross, S.; Gordon, G. J.; and Bagnell, J. A. 2011. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. arXiv:1011.0686.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Smolensky, P.; McCoy, R.; Fernandez, R.; Goldrick, M.; and Gao, J. 2022. Neurocompositional computing: From the

central paradox of cognition to a new generation of ai systems. *AI Magazine*, 43(3): 308–322.

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

Sutton, R. S.; Precup, D.; and Singh, S. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2): 181–211.

Szot, A.; Clegg, A.; Undersander, E.; Wijmans, E.; Zhao, Y.; Turner, J. M.; Maestre, N.; Mukadam, M.; Chaplot, D. S.; Maksymets, O.; Gokaslan, A.; Vondrus, V.; Dharur, S.; Meier, F.; Galuba, W.; Chang, A. X.; Kira, Z.; Koltun, V.; Malik, J.; Savva, M.; and Batra, D. 2021. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. *CoRR*, abs/2106.14405.

von Melchner, L.; Pallas, S. L.; and Sur, M. 2000. Visual behaviour mediated by retinal projections directed to the auditory pathway. *Nature*, 404: 871–876.

Wijmans, E.; Kadian, A.; Morcos, A.; Lee, S.; Essa, I.; Parikh, D.; Savva, M.; and Batra, D. 2020. DD-PPO: Learning Near-Perfect PointGoal Navigators from 2.5 Billion Frames. arXiv:1911.00357.

Yarbus, A. L. 1967. *Eye Movements and Vision*. New York: Plenum Press.

Zatka-Haas, P.; Steinmetz, N. A.; Carandini, M.; and Harris, K. D. 2021. Sensory coding and the causal impact of mouse cortex in a visual decision. *eLife*, 10: e63163.