A recurrent network model of planning explains hippocampal replay and human behavior

Kristopher T. Jensen[@1], **Guillaume Hennequin**[*1], **and Marcelo G. Mattar**[*2,3]

[1] Computational and Biological Learning Lab, Department of Engineering, University of Cambridge, Cambridge, UK
[2] Department of Cognitive Science, University of California, San Diego, USA
[3] Department of Psychology, New York University, New York, USA

[@] Corresponding author (ktj21@cam.ac.uk) [*] These authors jointly supervised the work

Abstract

When interacting with complex environments, humans can rapidly adapt their behavior to changes in task or context. To facilitate this adaptation, we often spend substantial periods of time contemplating possible futures before acting. For such planning to be rational, the benefits of planning to future behavior must at least compensate for the time spent thinking. Here we capture these features of human behavior by developing a neural network model where not only actions, but also planning, are controlled by prefrontal cortex. This model consists of a meta-reinforcement learning agent augmented with the ability to plan by sampling imagined action sequences drawn from its own policy, which we refer to as 'rollouts'. Our results demonstrate that this agent learns to plan when planning is beneficial, explaining the empirical variability in human thinking times. Additionally, the patterns of policy rollouts employed by the artificial agent closely resemble patterns of rodent hippocampal replays recently recorded in a spatial navigation task, in terms of both their spatial statistics and their relationship to subsequent behavior. Our work provides a new theory of how the brain could implement planning through prefrontal-hippocampal interactions, where hippocampal replays are triggered by – and in turn adaptively affect – prefrontal dynamics.

Introduction

Humans and other mammals have a unique ability to adapt rapidly to new information and changing environments. Such adaptation often involves spending extended and variable periods of time contemplating possible futures before taking an action (Callaway et al., 2022; van Opheusden et al., 2021). For example, we might take a moment to think about which route to take to work depending on traffic conditions. The next day, some roads might be blocked due to roadworks, requiring us to adapt and mentally review the available routes in a process of re-planning before leaving the house. Since thinking does not involve the acquisition of new information or interactions with the environment, it is perhaps surprising that it is so ubiquitous for human decision making. However, thinking allows us to perform more computations with the available information, which can lead to improved performance on downstream tasks (Bansal et al., 2022). Since physically interacting with the environment can consume time and other resources, or incur unnecessary risk, the benefits of planning often more than make up for the time that was lost to the planning process itself.

Despite a wealth of cognitive science research on the algorithmic underpinnings of planning (Solway and Botvinick, 2012; Callaway et al., 2022; Mattar and Daw, 2018; Mattar and Lengyel, 2022), little is known about the underlying neural mechanisms. This question has been difficult to address due to a scarcity of intracortical recordings during planning, and during contextual adaptation more generally. However, neuroscientists have begun to collect large-scale neural recordings during increasingly complex behaviors from the hippocampus and prefrontal cortex, brain regions known to be important for memory, decision making, and adaptation (Widloski and Foster, 2022; Pfeiffer and Foster, 2013; Gillespie et al., 2021; Wang et al., 2018; Samborska et al., 2022; Jadhav et al., 2016; Wu et al., 2017). These studies have demonstrated the importance of prefrontal cortex for generalizing abstract task structure across contexts (Wang et al., 2018; Samborska et al., 2022). Additionally, it has been suggested that planning could be mediated by the process of hippocampal forward replays (Pfeiffer and Foster, 2013; Widloski and Foster, 2022; Mattar and Daw, 2018; Agrawal et al., 2022; Foster, 2017; Jiang et al., 2022; Johnson and Redish, 2007). Despite these preliminary theories, little is known about how hippocampal replays could be integrated within the dynamics of downstream circuits to implement planning-based decision making and facilitate adaptive behavior (Jai and Frank, 2015). While prevailing theories of learning from replays generally rely on dopamine-mediated synaptic plasticity (Gomperts et al., 2015; Mattar and Daw, 2018; De Lavilléon et al., 2015), it is currently unclear whether this process could operate sufficiently fast to also inform online decision making.

It has recently been suggested that some forms of fast adaptation could result from recurrent meta-reinforcement learning (meta-RL; Wang et al., 2018, 2016; Duan et al., 2016). Such meta-RL models posit that adaptation to new tasks can be directly implemented by the recurrent dynamics of the prefrontal network. The dynamics themselves are learned through gradual changes in synaptic weights, which are modified over many different environments and tasks in a slow process of reinforcement learning. Importantly, such recurrent neural network (RNN)-based agents are able to adapt rapidly to a new task or environment after training by integrating their experiences into the hidden state of the RNN, with no additional synaptic changes (Wang et al., 2018, 2016; Duan et al., 2016; Zintgraf et al., 2019; Alver and Precup, 2021). However, previous models are generally only capable of making *instantaneous* decisions and thus do not have the ability to improve their choices by 'thinking' prior to taking an action. Wang et al. (2018) explored the possibility of allowing multiple steps of network dynamics before making a decision, but this additional computation was also predetermined by the experimenter and not adaptively modulated by the agent itself.

In this work, we propose a model that similarly combines slow synaptic learning with fast adaptation through recurrent dynamics in the prefrontal network. In contrast to previous work, however, this recurrent meta-learner can *choose* to momentarily forgo physical interactions with the environment and instead 'think' (Hamrick et al., 2017; Pascanu et al., 2017). This process of thinking is formalized as the simulation of sequences of imagined actions, sampled from the policy of the agent itself, which we refer to as 'rollouts' (Figure 1A). We introduce a flexible maze navigation task to study the relationship between the behavior of such RL agents and that of humans (Figure 1B). In this task, both human participants and RL agents (collectively 'subjects') have to discover the spatial location of an unknown goal in a novel environment, and they subsequently have to return to this goal from multiple different starting locations (Morris, 1981; Banino et al., 2018). Intriguingly, RL agents trained on this task learn to use rollouts to improve their policy and better generalize to previously unseen environments, and they selectively trigger rollouts in situations where humans also spend more time deliberating.

Additionally, we draw explicit parallels between the model rollouts and hippocampal replays through novel analyses of recent hippocampal recordings from rats performing a similar maze task (Widloski and Foster, 2022). We find that the content and behavioral effects of hippocampal replays in this dataset have a striking resemblance to the content and effects of policy rollouts in our computational model. Our work thus addresses two key questions from previous studies on hippocampal replays and planning. First, we show that a recurrent network can meta-learn when to plan instead of having to precompute a 'plan' in order to decide whether to use it (Mattar and Daw, 2018; Russek et al., 2022). Second, we propose a new theory of replay-mediated planning, which utilizes fast network dynamics for real-time decision making that could operate in parallel to slower synaptic plasticity (Gomperts et al., 2015). To formalize this second point, we provide a normative mathematical theory of how replays can improve decision making via feedback to prefrontal cortex by approximating policy gradient optimization (Sutton and Barto, 2018). We show that such an optimization process naturally arises in our RL agent trained for rapid adaptation and suggest that biological replays could implement a similar process of rollout-driven decision making (Figure 1C).

Our work provides new insights into the neural underpinning of 'thinking' by bridging the gap between recurrent meta-RL (Wang et al., 2018), machine learning research on adaptive computation (Hamrick et al., 2017; Graves, 2016; Banino et al., 2021), and theories of meta-cognition (Griffiths et al., 2015; Botvinick and Cohen, 2014; Botvinick et al., 2020). We link these idea to the phenomenon of hippocampal replays and provide a new theory of how forward replays can modulate behavior through recurrent interactions with prefrontal cortex.

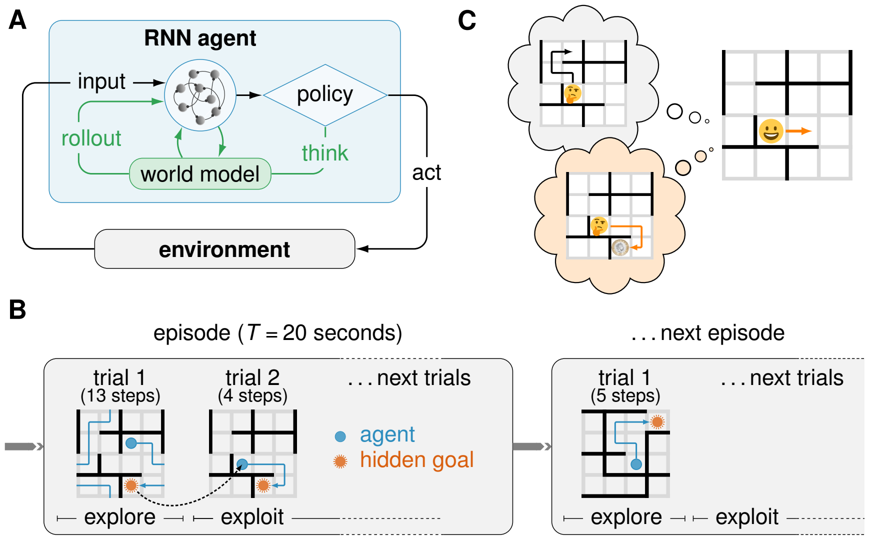

Figure 1: **Task and model schematics. (A)** The RL agent consisted of a recurrent neural network, which received information about the environment and executed actions in response. The primary output of the agent was a policy from which the next action was sampled. This action could either be to move in the environment in a given direction (up, down, left or right), or to 'plan' by using an internal world model to simulate a possible future trajectory (a 'rollout'). The agent was trained to maximize its average reward per episode and to predict (i) the upcoming state, (ii) the current goal location, and (iii) the value of the current state. When the agent decided to plan, the first two predictors were used in an open-loop planning process, where the agent iteratively sampled 'imagined' actions and predicted what the resulting state would be, and whether the goal had been (virtually) reached. The output of this planning process was appended to the agent's input on the subsequent time step (details in text). A physical action was assumed to take 400 ms and a rollout was assumed to take 120 ms. **(B)** Schematic illustrating the dynamic maze task. In each episode lasting $T = 20$ seconds, a maze and a goal location were randomly sampled. Each time the goal was reached, the subject received a reward and was subsequently "teleported" to a new random location, from which it could return to the goal to receive more reward. The maze had periodic boundaries, meaning that subjects could exit one side of the maze to appear at the opposite side. **(C)** Schematic illustrating how policy rollouts can improve performance by altering the momentary policy. An agent might perform a policy rollout leading to low value (top; black), which would *decrease* the probability of physically performing the corresponding sequence of actions. Conversely, a rollout leading to high value (bottom; orange) would *increase* the probability of the corresponding action sequence. Notably, these policy changes occur at the level of network dynamics rather than parameter updates.

Results

Humans think for different durations in different contexts

To characterize the behavioral signatures of planning, we recruited 94 human participants from Prolific to perform an online experiment. The experiment consisted of a maze navigation task in which the walls and goal location periodically changed, thus requiring rapid adaptation. The environment was a 4×4 grid with periodic boundaries, a set of impassable walls, and a single hidden reward location (Figure 1B; Methods). The task consisted of a succession of 'episodes', each lasting $T = 20$ seconds. At the beginning of each episode, both the wall configuration and the reward location were randomly initialized and remained fixed until the next episode. The initial position of the subject was also randomly sampled. Subjects first had to explore the maze by taking discrete steps in the cardinal directions until they found the hidden reward location. Upon finding this goal, subjects were immediately moved to a new random location, initiating a phase of exploitation during which they repeatedly had to return to the same goal (Figure 1B). We refer to a single instance of navigating from a random starting location to the goal as a 'trial'. To encourage good performance, human participants were paid a monetary bonus proportional to the average number of trials completed per episode (Methods; Figure S1), and the behavior of all subjects was recorded over 40 episodes.

We first examined human performance as a function of trial number within each episode, comparing the first exploration trial with subsequent exploitation trials. We found that participants exhibited a rapid 'one-shot' transition to goal-directed navigation after the initial exploration phase (Figure 2A, black). This was true even though each new maze was not seen before, and it is consistent with previous work demonstrating the ability of humans and animals to adapt rapidly to new information in a 'meta-learning' setting (Wang et al., 2018). We next investigated the time participants spent thinking during the exploitation phase. We estimated the 'thinking time' for each action as the posterior mean under a probabilistic model that decomposes the total response time for each action (Figure 2B; top) into the sum of the thinking time (Figure 2B; bottom) and a perception-action delay. The prior distribution over perception-action delays was estimated for each individual using a separate set of trials, where participants were explicitly cued with the optimal path and thus did not have to plan a route themselves (Methods; Figure S1). Since the first step within each trial required participants to parse their new position in the maze, a separate prior was fitted for the first action in a trial.

Participants exhibited a wide distribution of thinking times during the exploitation phase of the task (Figure 2B; bottom). To reveal any task-related structure in this variability, we partitioned thinking times by within-trial action number and by distance to goal (Figure 2C). We found that participants exhibited longer thinking times when further from the goal, consistent with planning of longer routes taking more time. Furthermore, subjects exhibited substantially longer thinking times for the first action of each trial (Figure S2), consistent with them having to initially plan a new route to the goal. These patterns confirm that the broad marginal distribution of thinking times (Figure 2B) does not simply reflect a noisy decision-making process or task-irrelevant distractions. On the contrary, variability in thinking time is an important feature of human behavior that reflects the variable moment-to-moment cognitive demands for decision making.

A recurrent network model of planning

To model the rapid adaptation and the detailed patterns of thinking times displayed by human subjects, we considered an RNN model trained in a meta-reinforcement learning setting (Figure 1A; Methods; Duan et al., 2016; Wang et al., 2016, 2018; see Supplementary Note for a more in-depth discussion and motivation of our modeling choices). The RL agent consisted of 100 gated recurrent units (GRUs; Cho et al., 2014; Figure S3) and was characterized by a time-varying internal activation state $\boldsymbol{h}_k$, which evolved dynamically according to

$$\boldsymbol{h}_k = \phi_\theta(\boldsymbol{x}_k, \boldsymbol{h}_{k-1}) \tag{1}$$

$$\boldsymbol{y}_k = \zeta_\theta(\boldsymbol{h}_k). \tag{2}$$

Here, θ denotes the set of all model parameters, $\boldsymbol{x}_k$ are momentary inputs to the RNN, and $\boldsymbol{y}_k$ are momentary network outputs computed from the current state $\boldsymbol{h}_k$, which was reset at the beginning of each episode. k indexes the evolution of the network dynamics and can in general be *different* from the wallclock time t in agents that have the ability to 'think' for variable periods of time (see below). Inputs consisted of the current agent location $\boldsymbol{s}_k$, the previous action taken a_{k-1} and associated reward signal r_{k-1}, the elapsed time t since the beginning of the episode, and the locations of all walls (Methods). Thus, while the reward location was hidden and had to be both discovered and memorized, the rest of the environment was fully observed. Outputs consisted primarily of a policy $\pi_\theta(a_k|\boldsymbol{h}_k)$, i.e. a set of probabilities associated with each possible action, which depended on the current hidden state of the RNN. At each step, an action a_k was sampled from this distribution and triggered changes in the environment ψ according to:

$$\boldsymbol{x}_{k+1}, \boldsymbol{s}_{k+1} = \psi(a_k, \boldsymbol{s}_k). \tag{3}$$

This yielded both a new location $\boldsymbol{s}_{k+1}$ of the agent and the new inputs $\boldsymbol{x}_{k+1}$, which were fed back to the agent on the subsequent iteration (Figure 1A). In addition to the policy, the output of the agent included a value function and predictions of the new location and current goal location.

As in standard RL settings, we quantified the performance of the agent in a given environment as the expected total reward,

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{k=1}^{K} r_k\right]. \tag{4}$$

Training proceeded by gradually adjusting the parameters θ to maximize the average $J(\theta)$ *across environments*, using a policy gradient algorithm (Methods; Sutton and Barto, 2018; Wang et al., 2018). In Equation 4, K refers to the total number of iterations in an episode, with each episode terminating once t exceeded the episode duration of $T = 20$ seconds as in the human data (Figure 1B).

Since our agent had no intrinsic notion of wallclock time, we considered each discrete action to consume $\Delta t = 400$ ms, meaning that there was time for 50 actions in a single RL episode. This was calculated to approximately match the number of actions taken in a typical RL episode to the human data. In this canonical formulation, the RL agent always takes an instantaneous action in response to a given set of inputs. It therefore does not have any ability to perform temporally extended planning, implying constant (zero) 'thinking time' in all situations. As a consequence, such a canonical meta-RL agent cannot explain the salient patterns of thinking times observed in human participants (recall Figure 2C). At first glance, temporally extended planning might also appear *unnecessary* in the RL agent, since it already has access to the current state, wall configuration, and reward information needed for decision making. However, this was also the case for our human participants, who chose to spend time thinking nonetheless. We hypothesized that the RL agent could similarly benefit from the ability to trade off time for additional *processing* of the available information in difficult tasks, where the agent has not learned a perfect policy (Hamrick et al., 2017; Pascanu et al., 2017).

To investigate the effect of such thinking for recurrent meta learners and account for the observed variability in human thinking times, we augmented the RL agent with the ability to perform temporally extended planning in the form of imagined policy rollouts. Specifically, we expanded the action space of the agent to give it the option of sampling a hypothetical trajectory from its own policy at any moment in time (a 'rollout'; Figure 1A; Hamrick et al., 2017; Pascanu et al., 2017). In other words, the agent was allowed to either perform a physical action, or to perform a mental simulation of its policy. If the agent chose to perform a rollout, a flattened array of the imagined action sequence was fed back to the network as additional inputs on the subsequent time step, together with an indication of whether or not the simulated action sequence reached the goal. These inputs in turn affected the policy by modulating $\boldsymbol{h}_k$ through a set of learnable input weights (Figure 1A). This is reminiscent of canonical RL algorithms that change their *parameters* θ to yield a new and improved policy on the basis of trajectories sampled from the current policy. In our formulation of planning, the agent's policy is instead induced by the *hidden state* $\boldsymbol{h}_k$, which can similarly be modulated on the basis of the imagined policy rollouts to improve performance.

Each rollout was terminated either upon reaching the goal, or after a maximum duration of 8 simulated actions (see Figure S3 for different network sizes and maximum planning horizons). Importantly, both the generation of a mental rollout and the corresponding success feedback relied on an *internal* model of the environment that was obtained from the agent itself. This internal model was trained alongside the RNN and the policy, by learning to predict the reward location and state transitions from the momentary hidden state of the RNN ($\boldsymbol{h}_k$) and the action taken (a_k; Methods; Figure S4). Thus, rollouts did not provide the agent with any privileged information that it did not already possess. Instead, they allowed the agent to trade off time for additional computational capacity – similar to thinking in humans and other animals. Furthermore, to capture the fact that mental simulation is faster than physical actions (Liu et al., 2019; Kurth-Nelson et al., 2016), we assumed each full rollout to consume only 120 ms. In other words, a single iteration of the network dynamics ($k \rightarrow k+1$ in Equation 1) incremented time by 120 ms if the agent chose to perform a rollout and 400 ms if the agent chose a physical action. This allowed the agent to perform many simulated actions in the time it would take to physically move only a short distance (Agrawal et al., 2022). Importantly, since an episode had a fixed duration of 20 seconds, choosing to perform more rollouts had a temporal opportunity cost by leaving less time for physical actions towards the goal.

Biologically, we interpret these mental simulations as prefrontal cortex (the RNN) interacting with the hippocampal formation (the world model), which allows the agent to simulate a sequence of state transitions from the current policy and evaluate their consequences. Importantly, while we endowed the agent with the ability to perform policy rollouts, we did not build in any prior knowledge about when, how, or how much they should be used. The agent instead had to learn this over the course of training on many different environments. Therefore, while the rollouts phenomenologically resembled hippocampal forward replays *by design*, our computational model allowed us to investigate (i) whether and how such rollouts can drive policy improvements, (ii) whether their temporal patterns can explain human response times, and (iii) whether biological replays appear to be implementing a similar computation.

The RL agent was trained by slowly adjusting its parameters θ over 8×10^6 episodes, sampled randomly

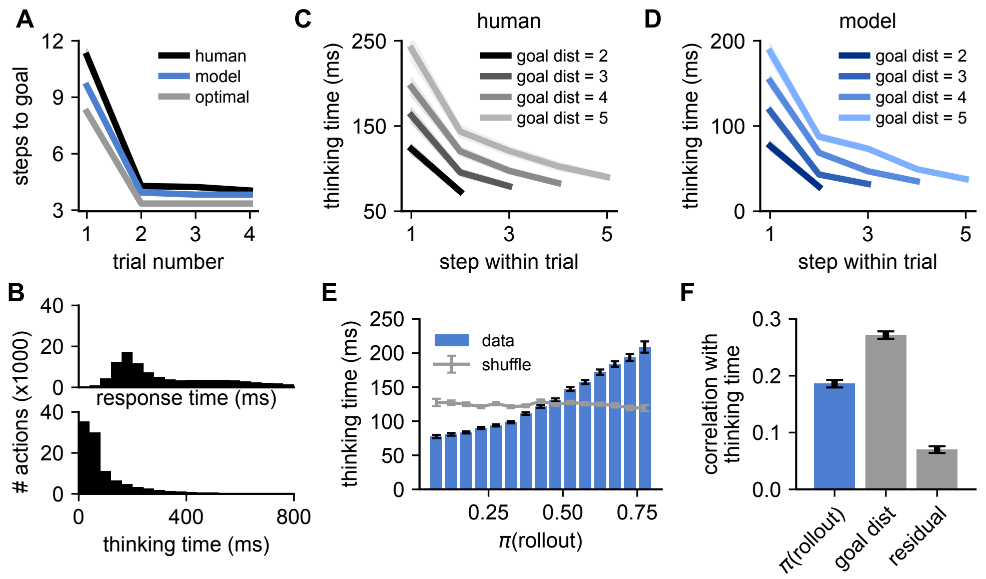

Figure 2: **Trained RL agents perform more rollouts in situations where humans spend longer thinking. (A)** Performance – quantified as the number of actions needed to reach the goal – as a function of trial number within each episode, computed for both human participants (black) and RL agents (blue). Shading indicates standard error of the mean across human participants ($n = 94$) or RL agents ($n = 5$) and mostly falls within the interval covered by the solid lines. Gray line indicates optimal performance, computed separately for exploration (trial 1) and exploitation (trials 2-4; Methods). **(B)** Distribution of human response times (top) and thinking times (bottom), spanning ranges on the order of a second (Methods). **(C)** Human thinking time as a function of the step-within-trial (x-axis) for different initial distances to the goal at the beginning of the trial (lines, legend). Shading indicates standard error of the mean across 94 participants. Participants spent more time thinking further from the goal and before the first action of each trial (Figure S2). **(D)** Model 'thinking times' separated by time-within-trial and distance-to-goal, exhibiting a similar pattern to human participants. To compute thinking times for the model, each rollout was assumed to last 120 ms as described in the main text. Shading indicates standard error of the mean across 5 RL agents. **(E)** Binned human thinking time as a function of the probability that the agent chooses to perform a rollout, π(rollout). Error bars indicate standard error of the mean within each bin. Gray horizontal line indicates a shuffled control, where human thinking times were randomly permuted before the analysis. **(F)** Correlation between human thinking time and the regressors (i) π(rollout) under the model, (ii) distance-to-goal, and (iii) π(rollout) after conditioning on distance-to-goal ('residual'; Methods). Bars and error bars indicate mean and standard error across human participants ($n = 94$).

from 2.7×10^8 possible environment configurations. This implied that the majority of environments seen at test time would be novel to the agent, requiring generalization across tasks. Parameter adjustments followed the gradient of a cost function that combined terms designed to (i) maximize the expected reward in Equation 4, (ii) learn the internal model by accurately predicting the reward location and state transitions, and (iii) minimize a standard entropy cost to encourage exploration (Methods; Wang et al., 2016). Importantly, parameters were frozen after training, and the agent adapted to the wall configuration and goal location of each new environment using only internal network dynamics (Wang et al., 2018; Duan et al., 2016).

Human thinking times correlate with agent rollouts

Having specified our computational model of planning, we analyzed its behavior and compared it to that exhibited by humans. We trained 5 copies of our RL agent to solve the same task as the human participants and found that the agents robustly learned to navigate the changing maze. Similar to humans, the trained agents exhibited a rapid transition from exploration to exploitation upon finding the reward, reaching near-optimal performance in both phases (Figure 2A, blue). This confirmed that these RNNs are capable of adapting to changing environments using only internal network dynamics with fixed parameters, corroborating previous work on recurrent meta-RL (Wang et al., 2018; Duan et al., 2016; Banino et al., 2018). The trained networks also used their capacity to perform rollouts, choosing to do so approximately 30% of the time. Importantly, there was temporal variability in the probability of performing a rollout, and the networks sometimes performed multiple successive rollouts between consecutive physical actions. When we queried the conditions under which the trained agents performed these rollouts, we found striking similarities with the pattern of human thinking times observed previously. In particular, the RL agent performed more rollouts earlier in a trial and further from the goal (Figure 2D) – situations where the human participants also spent more time thinking before taking an action (Figure 2C). On average, thinking times in the RL agent were approximately 50 ms lower than in humans. This difference could e.g. be due to (i) the choice of prior in the probabilistic model used to infer human thinking times, (ii) the agent having a better 'base policy' than humans, or (iii) the hyperparameters determining the temporal cost of planning.

To further study the relationship between rollouts and human 'thinking', we simulated the RL agent in the same environments as the human participants. We did this by clamping the physical actions of the agent to those taken by the participants, while still allowing it to sample on-policy rollouts (Methods). In this setting, the agent's probability of choosing to perform a rollout when encountering a new state, $\pi(\text{rollout})$, was a monotonically increasing function of human thinking time in the same situation (Figure 2E). The Pearson correlation between these two quantities was $r = 0.186 \pm 0.007$ (mean $\pm$ sem across participants), which was significantly higher than expected by chance (Figure 2F, '$\pi(\text{rollout})$'; chance level $r = 0 \pm 0.004$). An above-chance correlation between thinking times and $\pi(\text{rollout})$ of $r = 0.070 \pm 0.006$ persisted after conditioning on the distance-to-goal (Figure 2F, 'residual'), which was also correlated with thinking times ($r = 0.272 \pm 0.006$). The similarity between planning in humans and RL agents thus extends beyond this salient feature of this task, including an increased tendency to plan on the first step of a trial (Figure S2).

In addition to the similarities during the *exploitation* phase, a significant correlation was also observed between human thinking time and $\pi(\text{rollout})$ during *exploration* ($r = 0.098 \pm 0.008$). In this phase, both humans and RL agents spent more time thinking during later stages of exploration (Figure S5). Model rollouts during exploration corresponded to planning towards an imagined goal from the posterior over goal locations, which becomes narrower as more states are explored (Figure S5). This finding suggests that humans may similarly engage in increasingly goal-directed behavior as the goal posterior becomes narrower over the course of exploration. Taken together, our results show that a meta-reinforcement learning agent, endowed with the ability to perform rollouts, learns to do so in situations similar to when humans appear to plan. This provides a putative normative explanation for the variability in human thinking times observed in the dynamic maze task.

Rollouts improve the policy of the RL agent

In the previous section, we saw that an RL agent can learn to use policy rollouts as part of its decision making process, and that the timing and number of rollouts correlates with variability in human thinking times. In this section, we aim to understand *why* the agent chooses to perform rollouts and *how* they guide its behavior. To do this, we considered the agent right after it first located the goal in each episode (i.e., at the first time step of trial 2; Figure 1B) and forced it to perform a pre-defined number of rollouts, which we varied. We then quantified the number of actions that the agent took to return to the goal while preventing any further rollouts during this return phase (Methods).

The average number of actions needed to reach the goal decreased monotonically as the number of forced rollouts increased up to at least 15 rollouts (Figure 3A). Interestingly, this was the case despite the unperturbed behavior of the agent rarely including more than a few consecutive rollouts (Figure 2D), suggesting that the agent learned a robust algorithm for policy optimization on the basis of such rollouts (Schrittwieser et al., 2020; Hamrick et al., 2017). The increase in performance with rollout number was also associated with a concomitant decrease in policy entropy (Figure 3B). Thus, performing more rollouts both improved performance and increased the agent's confidence in its actions (Methods). These findings confirm that the agent successfully learned to use policy rollouts to optimize its future behavior. However, the question remains of whether this policy improvement is appropriately balanced with the temporal opportunity cost of performing a rollout.

In general, performing a rollout is beneficial in situations where the policy improvement resulting from the rollout is greater than the temporal cost of 120

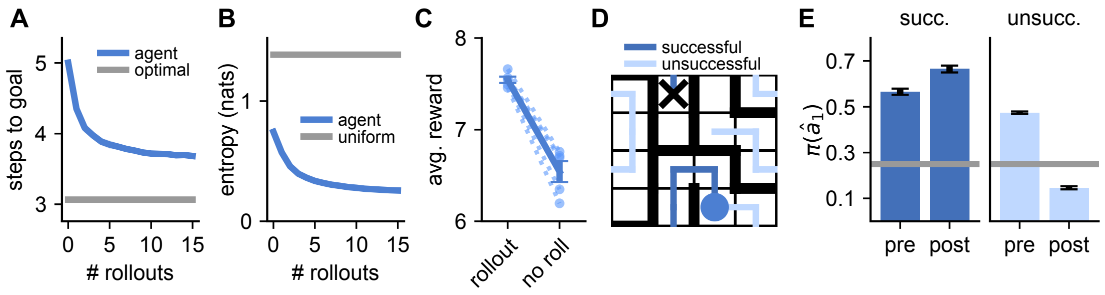

Figure 3: **Rollouts improve the network policy.** **(A)** Performance on trial 2 as a function of the number of rollouts enforced at the beginning of the trial. Performance was quantified as the average number of steps needed to reach the goal in the absence of further rollouts. Gray horizontal line indicates optimal performance. **(B)** Policy entropy as a function of the number of rollouts enforced at the beginning of trial 2. The entropy was computed after re-normalizing the policy over the four 'physical' actions, and the horizontal gray line indicates the entropy of a uniform policy. **(C)** Original performance of the RL agent (left) and its performance when re-normalizing the policy over physical actions to prevent any rollouts (right). Performance was quantified as the average number of rewards collected per episode, and dashed lines indicate individual RL agents, while the solid line indicates mean and standard error across agents. **(D)** Schematic showing an example of a 'successful' (dark blue) and an 'unsuccessful' (light blue) rollout from the same physical location (blue circle). Black cross indicates the goal location (not visible to the agent or human participants). **(E)** Probability of taking the first simulated action of the rollout, $\hat{a}_1$, before ($\pi^{\text{pre}}(\hat{a}_1)$) and after ($\pi^{\text{post}}(\hat{a}_1)$) the rollout. This was evaluated separately for successful (left) and unsuccessful (right) rollouts. $\pi^{\text{pre}}(\hat{a}_1)$ was above chance (gray line) in both cases and increased for successful rollouts, while it decreased for unsuccessful rollouts. Error bars represent standard error across five independently trained agents. The magnitude of the change in $\pi(\hat{a}_1)$ for successful and unsuccessful rollouts depended on the planning horizon (Figure S3).

ms of performing the rollout. To investigate whether the agent learned to trade off the cost and benefit of rollouts (Hamrick et al., 2017; Pascanu et al., 2017; Agrawal et al., 2022), we computed the performance of the agent in a surrogate environment where rollouts were not allowed. In this setting, each action was instead sampled from the distribution over physical actions only (Methods). When preventing rollouts in this way, the agent only collected 6.54 ± 0.11 rewards per episode compared to 7.54 ± 0.03 in the presence of rollouts (mean $\pm$ standard error across agents), confirming that it used rollouts to increase expected reward (Figure 3C). To investigate whether the temporal structure of rollouts described in Figure 2 was important for this performance improvement, we performed an additional control, where the number of rollouts was kept fixed for each environment, but their occurrence was randomized in time. In this case, performance dropped to 6.75 ± 0.04 rewards per episode, confirming that the RL agent chose to use rollouts specifically when they improved performance.

To further dissect the effect of rollouts on agent behavior, we classified each rollout, $\hat{\tau}$ (a sequence $\{\hat{a}_1, \hat{a}_2, \ldots\}$ of rolled-out actions), as being either 'successful' if it reached the goal according to the agent's internal world model, or 'unsuccessful' if it did not (Figure 3D). We hypothesized that the policy improvement observed in Figure 3A could arise from *upregulating* the probability of following a successful rollout and *downregulating* the probability of following an unsuccessful rollout. To test this hypothesis, we enforced a single rollout after the agent first found the reward and analyzed the effect of this rollout on the policy, separating the analysis by successful and unsuccessful rollouts. Importantly, we could compare the causal effect of rollout success by matching the history of the agent and performing rejection sampling from the rollout process until either a successful or an unsuccessful rollout had occurred (Methods). Specifically, we asked how the rollout affected the probability of taking the first rolled-out action, $\hat{a}_1$, by comparing the value of this probability before ($\pi^{\text{pre}}(\hat{a}_1)$) and after ($\pi^{\text{post}}(\hat{a}_1)$) the rollout. $\pi^{\text{pre}}(\hat{a}_1)$ was slightly higher for successful rollouts than unsuccessful rollouts, with both types of rollouts exhibiting a substantially higher-than-chance probability – a consequence of the model rollouts being drawn 'on-policy' (Figure 3E). However, while successful rollouts *increased* $\pi(\hat{a}_1)$, unsuccessful rollouts *decreased* $\pi(\hat{a}_1)$ (Figure 3E). This finding demonstrates that the agent combines the spatial information of a rollout with knowledge about its consequences, based on its internal world model, to guide future behavior.

Hippocampal replays resemble policy rollouts

In our computational model, we designed policy rollouts to take the form of spatial trajectories that the agent could subsequently follow, and to occur only when the agent was stationary. These two properties are also important signatures of forward hippocampal replays – patterns of neural activity observed using electrophysiological recordings from rodents during spatial navigation (Pfeiffer and Foster, 2013; Widloski and Foster, 2022; Gillespie et al., 2021). Our model therefore allowed us to investigate whether forward replay in biological agents serve a similar function during decision making to the function of policy rollouts in our RL agent. Additionally, since we have direct access to the agent's policy and how it changes after a replay, our computational model can provide insights into the apparently conflicting data and contradictory viewpoints in the literature regarding the role of hippocampal replays. In particular, some studies have found a significant correlation between forward replay and subsequent behavior (Pfeiffer and Foster, 2013; Foster, 2017; Widloski and Foster, 2022), arguing that such a correlation suggests a role of forward replay for planning. On the contrary, other studies have found that forward replays do not always resemble subsequent behavior (Gillespie et al., 2021; Krause and Drugowitsch, 2022; Wu et al., 2017), challenging the interpretation of forward replay as a form of planning. Our model offers a potentially conciliatory explanation, predicting that the correlation between forward replay and subsequent behavior can be positive or negative, depending on the replayed trajectory (Figure 3E; Jai and Frank, 2015; Antonov et al., 2022).

To investigate whether there is evidence for such replay-based modulation of animal behavior, we reanalyzed a recently published hippocampal dataset from rats navigating a dynamic maze very similar to the task in Figure 1B (Widloski and Foster, 2022). Our goal was to compare the recorded replay events to the policy rollouts exhibited by the RL agent, considering both the statistical properties of the replays themselves and how they relate to subsequent behavior. In this rodent experiment, animals had to repeatedly return to an initially unknown 'home' location, akin to the goal in our task (Figure S6). Both this home location and the configuration of the maze changed between sessions. Whilst the behaving animals could not be 'teleported' between trials as in our task, rats instead had to navigate to an unknown rewarded 'away' location selected at random after each 'home' trial. These 'away' trials served as a useful control since the animals did not know the location of the rewarded well at the beginning of the trial.

We studied replay events detected in hippocampal recordings made with tetrode drives during the maze task ($n \in [187, 333]$ simultaneously recorded neurons per session; Figure S6C). To detect replays, we followed Widloski and Foster (2022) and first trained a Bayesian decoder to estimate the animal's position on a discretized grid from the neural data during epochs when the animal was moving. We then applied this decoder during epochs when the animal was stationary at a reward location before initiating a new trial and defined replays as consecutive sequences of at least three adjacent decoded grid locations (Figure 4A; Figure S6; see Methods for details).

Similar to previous work (Widloski and Foster, 2022), we found that the hippocampal replays avoided passing through walls to a greater extent than expected by chance (Figure 4B; $p < 0.001$, permutation test). This finding suggests that hippocampal replays are shaped by a rapidly updated internal model of the environment, similar to how forward rollouts in our RL agent are shaped by its internal world model (Figure 1A). Additionally, the goal location was overrepresented in the hippocampal replays, consistent with the assumption of on-policy rollouts in the RL agent (Figure 4C; $p < 0.001$, permutation test; Widloski and Foster, 2022).

Inspired by our findings in the RL agent, we proceeded to investigate whether a replayed action was more likely to be taken by the animal if the replay was successful than if it was unsuccessful. Here, we defined a 'successful' replay as one which reached the goal location without passing through a wall (Figure 4A). Consistent with the RL model, we found that the first simulated action in the replay agreed with the next physical action more often for successful replays than for unsuccessful replays (Figure 4D, black; $p < 0.001$, permutation test). Such an effect was not observed in the 'away' trials (Figure 4D, gray; $p = 0.129$, permutation test), where the animals had no knowledge of the reward location and therefore could not know what constituted a successful replay. These findings are consistent with the hypothesis that successful replays should increase the probability of taking the replayed action, while unsuccessful replays should decrease this probability.

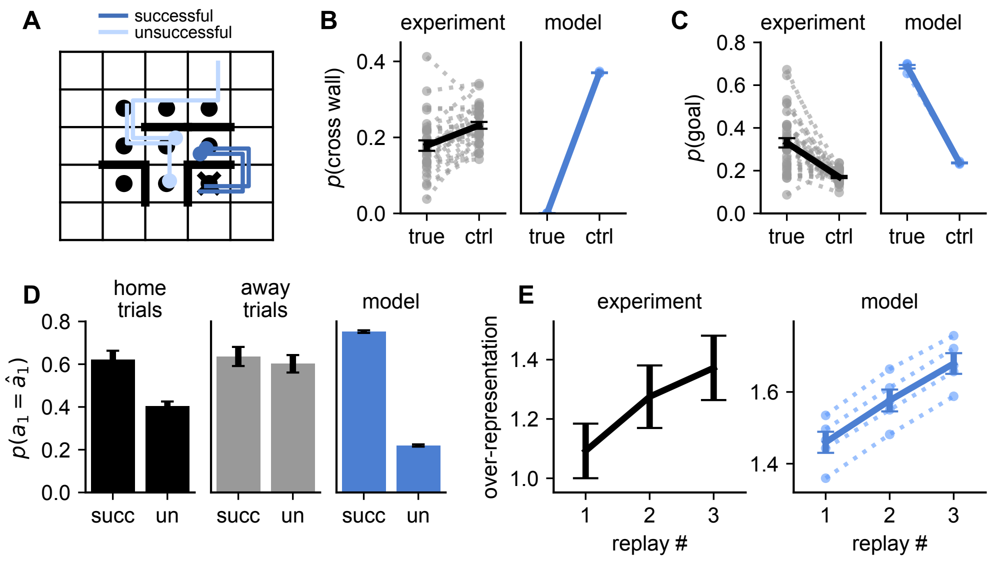

Figure 4: **Hippocampal replays resemble model rollouts.** **(A)** Illustration of experimental task structure and example replays (Widloski and Foster, 2022). Each episode had a different wall configuration and a randomly sampled home location (cross). Between each 'home' trial, the animal had to move to an 'away' location, which was sampled anew on each trial (black circles). Colored lines indicate example replay trajectories originating at the blue dots. Replays were detected during the stationary periods at the away locations before returning to the home location and classified according to whether they reached the home location (dark vs. light blue lines). **(B)** Fraction of replay transitions that pass through a wall in the experimental (black) and model (blue) data. Control values indicate the fraction of wall crossings in re-sampled environments with different wall configurations. Dashed lines indicate individual animals or RL agents, and solid lines indicate mean and standard error across animals or RL agents. **(C)** Fraction of replays that pass through the goal location in experimental (black) and model (blue) data. Control values indicate the average fraction of replays passing through a randomly sampled non-goal location. Dashed and solid lines are as in (B). See Figure S7 for an analogous analysis of the away trials, where the goal was unknown. **(D)** Probability of taking the first replayed action, $p(a_1 = \hat{a}_1)$, for successful and unsuccessful replays during home trials (left; black), away trials (center; gray), and in the RL agent (right; blue). Bars and error bars indicate mean and standard error across sessions or RL agents. **(E)** Over-representation of successful replays during trials with at least three replays in the experimental data (left) and RL agents (right). The over-representation increased with replay number; an effect not seen in the away trials (Figure S7). Over-representation was computed by dividing the success frequency by a reference frequency computed for randomly sampled alternative goal locations. Error bars indicate standard error across replays pooled from all animals (left) or standard error across five independently trained agents (right; dashed lines).

In the RL agent, we have direct access to the momentary policy and could therefore quantify the causal effect of a replay on behavior (Figure 3E). However, in the biological circuit, we cannot know whether the increased probability of following the first action of a successful replay is because the replay altered the policy (as in the RL agent), or whether the replay reflects a baseline policy that was already more likely to reach the goal prior to the replay. To circumvent this confound, we analyzed consecutive replays while the animal remained stationary. If our hypotheses hold, that (i) hippocampal replays resemble on-policy rollouts of an imagined action sequence, and (ii) performing a replay improves the policy, then consecutive replays should become increasingly successful even *in the absence* of any behavior between the replays.

To test this prediction, we considered trials where the animal performed a sequence of at least 3 replays at the 'away' location before moving to the 'home' location. We then quantified the fraction of replays that were successful as a function of the replay index within the sequence, after regressing out the effect of time (Methods; Ólafsdóttir et al., 2017). We expressed this quantity as the degree to which the true goal was over-represented in the replay events by dividing the fraction of successful replays by a baseline calculated from the remaining non-goal locations, such that an over-representation of 1 implies that a replay was no more likely to be successful than expected by chance. Compellingly, this over-representation increased with each consecutive replay during the home trials (Figure 4E; left), and both the second and third replays exhibited substantially higher over-representation than the first replay ($p = 0.068$ and $p = 0.009$ respectively; permutation test; Methods). Such an effect was not seen during the away trials, where the rewarded location was not known to the animal (Figure S7).

These findings are consistent with a theory in which replays represent on-policy rollouts that are used to iteratively refine the agent's policy, which in turn improves the quality of future replays – a phenomenon also observed in the RL agent (Figure 4E, right). In the RL agent, this effect could arise in part because the agent is less likely to perform an additional rollout after a successful rollout than after an unsuccessful rollout (Figure S8). To eliminate this confound, we drew two samples from the policy each time the agent chose to perform a rollout, and we used one sample to update the hidden state of the agent, while the second sample was used to compute the goal over-representation (Methods). Such decoupling is not feasible in the experimental data, since we cannot read out the 'policy' of the animal. This leaves open the possibility that the increase in goal over-representation with consecutive biological replays is in part due to a reduced probability of performing an additional replay after a successful replay. However, we note that (i) the rodent task was not a 'reaction time task', since a 5-15 s delay was imposed between each trial. This makes a causal effect of replay success on the total number of replays less likely. (ii) if such an effect does exist, that is in itself consistent with a theory in which hippocampal replays guide planning.

RL agents use rollouts to optimize their hidden state

We have now seen that both biological and artificial agents appear to use policy rollouts to influence behavior in a way that depends on the *content* of the rollout. However, it remains to be understood (i) whether such an algorithm formally increases the expected reward, and (ii) how it is implemented mechanistically – a question we can address in the trained RL agent. In this section, we show that our theory has a firm theoretical grounding and makes quantitative predictions about the neural implementation of planning in PFC. Previously, we showed that the agent up- or downregulated the probability $p(\tau = \hat{\tau})$ of actually performing a rolled-out sequence $\hat{\tau}$ depending on the 'goodness' of the rollout (Figure 3E). This is reminiscent of canonical policy-gradient RL algorithms. These algorithms consider putative on-policy action sequences τ and apply *parameter updates* that cause $p(\tau)$ to increase under the agent's policy if τ led to more reward than expected, and to decrease otherwise. In our trained agent, adaptation to each new maze does not involve modifications of the fixed network parameters but instead occurs through changes to the *hidden state* $\boldsymbol{h}_k$. We therefore hypothesized that the performance improvements resulting from policy rollouts (Figure 3A; Figure 4E) were achieved through iterative modifications of $\boldsymbol{h}_k$ that approximated policy gradient ascent on the expected future reward in the episode as a function of $\boldsymbol{h}_k$ (Figure 5A).

To test this hypothesis, we considered each rollout performed by the RL agent and computed both (i) the actual hidden state update performed by the RL agent on the basis of this rollout, and (ii) the expected hidden state update computed by applying the policy gradient algorithm to the same rollout (Figure 5B; Methods). Our theory predicts that rollouts should change $\boldsymbol{h}_k$ in a way that increases $p(\tau = \hat{\tau})$ if the rollout is better than some baseline and decreases $p(\tau = \hat{\tau})$ otherwise. Since we do not know the baseline, we performed our analysis by taking the *derivative* of the hidden state change with respect to the expected reward from physically following $\hat{\tau}$, $R_{\hat{\tau}}$, which is independent of the baseline (Methods). This allowed us to define (i) a quantity $\boldsymbol{\alpha}^{\text{PG}} := \frac{\partial \Delta \boldsymbol{h}^{\text{PG}}}{\partial R_{\hat{\tau}}}$ that predicts how the hidden state *should* change as a function of $R_{\hat{\tau}}$ in the policy gradient formulation, and (ii) the corresponding quantity $\boldsymbol{\alpha}^{\text{RNN}} := \frac{\partial \Delta \boldsymbol{h}^{\text{RNN}}}{\partial R_{\hat{\tau}}}$ that indicates how the hidden state *actually* changed as a function of the content of the rollout. If the agent performs approximate policy gradient ascent in hidden state space, $\boldsymbol{\alpha}^{\text{RNN}}$ should be aligned with $\boldsymbol{\alpha}^{\text{PG}}$.

To investigate whether the response of the RNN to a rollout was consistent with this theory, we began by considering the effect on its hidden state of the first action in the rollout, $\hat{a}_1$. We did this by querying the alignment between (i) $\boldsymbol{\alpha}^{\text{RNN}}$ computed across rollouts from 1,000 episodes, and (ii) $\boldsymbol{\alpha}_1^{\text{PG}}$ computed from the same rollouts when considering only the probability of executing $\hat{a}_1$. To visualize this alignment, we performed PCA on $\{\boldsymbol{\alpha}_1^{\text{PG}}\}$ from all rollouts and projected both $\boldsymbol{\alpha}_1^{\text{PG}}$ and $\boldsymbol{\alpha}^{\text{RNN}}$ into this low-dimensional subspace. We then computed the average of each of these two quantities for each simulated action $\hat{a}_1 \in \{\text{left}, \text{right}, \text{up}, \text{down}\}$. We

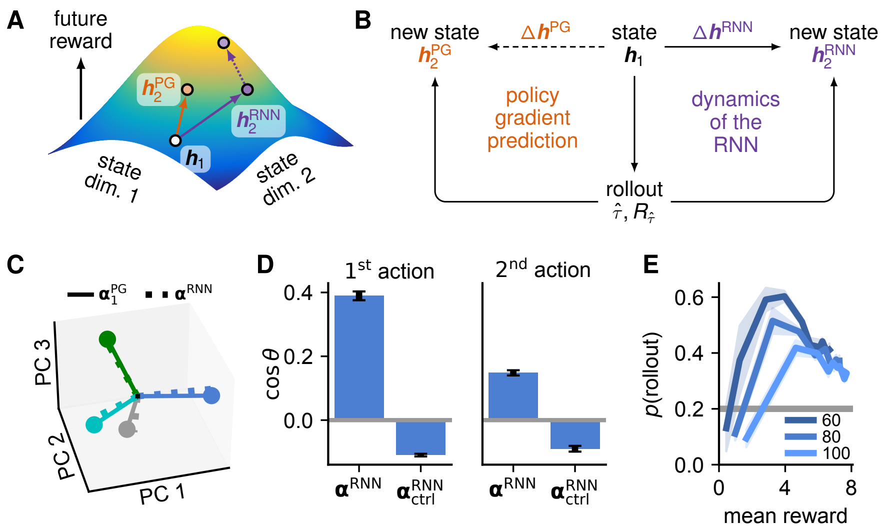

Figure 5: **Rollouts implement a hidden state optimization. (A)** The hidden state $\boldsymbol{h}_k$ of the RNN induces a policy with an expected future reward for the current episode. Rollouts can improve performance by shifting $\boldsymbol{h}$ to a region of state space with higher reward ($\boldsymbol{h}_1 \rightarrow \boldsymbol{h}_2^{\text{RNN}}$). The policy gradient algorithm estimates the direction of steepest ascent of the expected reward ($\boldsymbol{h}_1 \rightarrow \boldsymbol{h}_2^{\text{PG}}$). **(B)** We wanted to compare this theoretical hidden state update $\Delta\boldsymbol{h}^{\text{PG}} := \boldsymbol{h}_2^{\text{PG}} - \boldsymbol{h}_1$ to the empirical hidden state update $\Delta\boldsymbol{h}^{\text{RNN}} := \boldsymbol{h}_2^{\text{RNN}} - \boldsymbol{h}_1$ actually performed by the network dynamics on the basis of a rollout $\hat{\tau}$ and its associated reward $R_{\hat{\tau}}$. **(C)** A latent space was defined by performing PCA on $\boldsymbol{\alpha}_1^{\text{PG}}$ – the effect of $R_{\hat{\tau}}$ on $\boldsymbol{h}_k$ under the policy gradient algorithm. Solid lines and circles indicate the normalized average $\boldsymbol{\alpha}_1^{\text{PG}}$ for each of the four possible simulated actions ($\hat{a}_1$; colors). Dashed lines indicate the normalized average value of $\boldsymbol{\alpha}^{\text{RNN}}$ for the corresponding action, which is aligned with $\boldsymbol{\alpha}_1^{\text{PG}}$ in accordance with the theory. The first 3 PCs capture 100% of the variance in $\boldsymbol{\alpha}_1^{\text{PG}}$, since the policy is normalized and therefore only has three degrees of freedom. **(D)** Average cosine similarity between $\boldsymbol{\alpha}^{\text{RNN}}$ and $\boldsymbol{\alpha}_1^{\text{PG}}$, quantified in the space spanned by the top 3 PCs of $\boldsymbol{\alpha}_1^{\text{PG}}$ (see text for details). $\boldsymbol{\alpha}^{\text{RNN}}$ was computed using the true input, while $\boldsymbol{\alpha}_{\text{ctrl}}^{\text{RNN}}$ was computed after altering the feedback from the rollout to falsely inform the agent that it had simulated a different action $\hat{a}_{1,\text{ctrl}} \neq \hat{a}_1$. This confirms that the observed alignment is mediated by the input from the rollout. Left panel considers the effect of $R_{\hat{\tau}}$ on the first action ($\boldsymbol{\alpha}_1^{\text{PG}}$) and right panel considers the effect of $R_{\hat{\tau}}$ on the second action ($\boldsymbol{\alpha}_2^{\text{PG}}$). **(E)** We trained networks of different sizes (legend) and quantified both their performance (x-axis) and frequency of performing a rollout (y-axis) over the course of training (Figure S9). To reach a given performance, we found that smaller networks relied *more* on rollouts, suggesting that the RL agents learn to plan in part because they are capacity limited. Additionally, the agents learned to rely less on rollouts late in training as they became increasingly good at the task, suggesting that they also plan because they are data limited.

found that the average value of $\boldsymbol{\alpha}^{\text{RNN}}$ was strongly aligned with the average value of $\boldsymbol{\alpha}_1^{\text{PG}}$ for each action (Figure 5C), consistent with the theory outlined above. Importantly this means that $R_{\hat{\tau}}$ has *different* effects on the policy depending on the replayed trajectory $\hat{\tau}$. In other words, the spatial content of the rollout dynamically modulates the way in which the reward signal from the rollout affects the hidden state and policy of the agent.

To quantify the overlap between $\boldsymbol{\alpha}^{\text{RNN}}$ and $\boldsymbol{\alpha}_1^{\text{PG}}$ on a rollout-by-rollout basis, we computed the average cosine similarity d between $\boldsymbol{\alpha}^{\text{RNN}}$ and $\boldsymbol{\alpha}_1^{\text{PG}}$ across all rollouts. This overlap was substantially larger than zero ($d = 0.39 \pm 0.01$ mean $\pm$ sem; Figure 5D, left). When instead computing the overlap with $\boldsymbol{\alpha}_{\text{ctrl}}^{\text{RNN}}$ computed after changing the feedback input to falsely inform the agent that it simulated a different action $\hat{a}_{1,\text{ctrl}} \neq \hat{a}_1$, the corresponding value was $d = -0.11 \pm 0.004$. This confirms that $\boldsymbol{h}_k$ is optimized by incorporating the specific feedback input obtained from the rollout, and the negative sign reflects anti-correlations due to the policy being a normalized distribution over actions. For these analyses, we only considered the first simulated action $\hat{a}_1$. When instead querying the effect of the rollout on subsequent actions in $\hat{\tau}$, we found that the feedback input was also propagated through the network dynamics to these later actions (Figure 5D, right). These analyses confirm that policy rollouts consistently move the hidden state of the agent in the direction of the policy gradient.

Discussion

We have developed a new theory of planning in the prefrontal-hippocampal network, implemented as a recurrent neural network model and instantiated in a spatial navigation task requiring multi-step planning (Figure 1). Our model consists of a recurrent meta-reinforcement learning agent augmented with the explicit ability to plan using policy rollouts. We showed that this model provides a compelling account of human behavior in our task, where it explains the structure observed in human thinking times (Figure 2). These results suggest that planning using mental rollouts could constitute a major component underlying the striking human ability to adapt rapidly to new information and changing environments, where it allows agents to refine their behavior without incurring the potentially large cost of overtly executing suboptimal actions. Since mental simulation is generally faster and more efficient than physically interacting with the world (Vul et al., 2014), this allows agents to improve their overall performance despite the temporal opportunity cost of such simulation (Figure 3; Agrawal et al., 2022; Hamrick et al., 2017).

Our theory also suggests an important role of hippocampal replays during sequential decision making. By re-analyzing recordings from the rat hippocampus during a navigation task, we found that patterns of hippocampal replays and their relationship to behavior resembled the rollouts used by our model (Figure 4). These results suggest that hippocampal forward replays could be a manifestation of a planning process, and that the mechanistic insights derived from our model could generalize to biological circuits. In particular, we hypothesize that forward replays should have different effects on subsequent behavior depending on whether they lead to high-value or low-value states (Figure 3; Wu et al., 2017). This hypothesis is consistent with previous models, where hippocampal replay is used to update state-action values that shape future behavior (Mattar and Daw, 2018). We suggest that forward replay implements planning through feedback to prefrontal cortex that drives a 'hidden state optimization' reminiscent of recent models of motor preparation (Figure 5; Kao et al., 2021). This differs from prior work in the reinforcement learning literature, since our model does not involve arbitration between model-free and model-based policies computed separately (Daw et al., 2005; Geerts et al., 2020). Instead, model-based computations iteratively update a single policy that can be used for decision making at different stages of refinement.

Neural mechanisms of planning and decision making

Our model raises several interesting hypotheses about neural dynamics in hippocampus and prefrontal cortex and how these dynamics affect behavior. One is that hippocampal replays should causally affect the behavior of an animal as also suggested in previous work (Foster, 2017; Pfeiffer and Foster, 2013; Widloski and Foster, 2022). However, as noted previously (Figure 4), this has been notoriously difficult to test in experiments due to the confound of how the behavioral intentions of the animal itself affect the content of hippocampal replays (Foster, 2017). Perhaps more interestingly, we predict that hippocampal forward replays should directly drive a change in PFC representations, consistent with previous work showing coordinated activity between hippocampus and PFC during sharp-wave ripples (Jadhav et al., 2016). Crucially, we also predict *how* PFC representations should change during planning depending on the spatial content and expected reward of a replay. These predictions could be investigated in experiments that record neural activity simultaneously from hippocampus and PFC, where both the timing and qualitative change in PFC representations can be related to the occurrence of replays in hippocampus.

To enable more detailed mechanistic predictions, our model could be extended in several ways. First, we have modeled the prefrontal network as a single fully connected network. In contrast, the brain relies on several connected but distinct circuits, all of which serve specialized functions that together give rise to the representations and dynamics driving human behavior. To understand these collective dynamics, it will therefore be interesting to extend our approach to modular models inspired by the architecture of multi-area networks. Second, our implementation of rollouts in the agent took the form of an abstract simulation process, where the underlying neural dynamics were not explicitly modeled. To better understand the mechanisms through which PFC interacts with other brain areas during planning, it will be important to model the whole rollout process as multi-area neural dynamics. Finally, while we propose a role of hippocampal replays in shaping immediate behavior via recurrent network dynamics, this is compatible with replays also having other functions, such as memory consolidation (van de Ven et al., 2016; Carr et al., 2011) or dopamine-driven synaptic plasticity over longer timescales (Gomperts et al., 2015; De Lavilléon et al., 2015).

Alternative planning algorithms

Planning in the RL agent was carried out explicitly in the space of observations. While this was already an abstract representation rather than pixel-level input, it could be interesting to explore planning in a latent space optimized e.g. to predict future observations (Zintgraf et al., 2019) or future policies and value functions (Ho et al., 2022; Schrittwieser et al., 2020). These ideas have proven useful in the machine learning literature, where they allow models to ignore details of the environment not needed to make good decisions, and it is plausible that the internal model of humans similarly does not include such task-irrelevant details. We also assumed that the planning process itself was 'on policy' – that is, the policy that was used to sample actions in the planning loop was identical to the policy used to act in the world. Although there is some support from the hippocampal replay data that forward replays are related to the 'policy' (e.g. wall avoidance and goal over-representation; Figure 4), there is in theory nothing that prevents the planning policy from differing arbitrarily from the action policy. In fact, the planning policy could even be explicitly optimized to yield good *plans* rather than re-using a policy optimized to yield good *behavior* (Pascanu et al., 2017). Such off-policy hippocampal sequence generation has also formed the basis of other recent theories of the role of hippocampus in planning and decision making (McNamee et al., 2021; Mattar and Daw, 2018). In this case, the policy gradient view of rollouts still provides a natural language for formalizing the planning process, since numerous off-policy extensions of the canonical policy gradient algorithm exist (Peshkin and Shelton, 2002; Jie and Abbeel, 2010).

Why do we spend time thinking?

Finally, while both humans and our RL agents made extensive use of planning, it is worth noting that mental simulation does not generate any new information about the world. In theory, it should therefore be possible to make equally good 'reflexive' decisions given enough computational power. This raises the question of why we rely on planning in the first place – in other words, what is the reason that decision making often takes time rather than being instantaneous? One possible reason could be that our decision making system is capacity limited, such that it does not have enough computational power to generate the optimal policy (Russek et al., 2022). In our computational model, this is supported by the observation that agents consisting of smaller RNNs tend to perform *more* rollouts than larger agents (Figure 5E). Alternatively, we could be data limited, meaning that we have not received enough training to learn the optimal policy. This also has support in our computational model, where networks of all sizes perform many rollouts early in training, when they have only seen a small amount of data, and gradually transition to a more reflexive policy that relies less on rollouts (Figure 5E; Figure S9).

We hypothesize that data limitations are a major reason for the use of temporally extended planning in animals. In particular, we reason that learning the instantaneous mapping from states to actions needed for reflexive decisions would require a prohibitive amount of training data, which is generally not available for real-life scenarios. Indeed, training our meta-reinforcement learner required millions of episodes, while humans were immediately capable of solving the maze task from only a simple task description and demonstration. Such rapid learning could be due in part to the use of temporally extended planning algorithms as a form of 'canonical computation' that generalizes across tasks. If this is the case, we would be able to rely on generic planning algorithms acquired over the course of many previous tasks in order to solve a new task. When combined with a new task-specific transition function learned from relatively little experience or inferred from sensory inputs, planning would facilitate data-efficient reinforcement learning by allowing the agent to trade off processing time for a better policy (Schrittwieser et al., 2020). This is in contrast to our current model, which had to learn from scratch both the structure of the environment *and* how to use rollouts to shape its behavior. Importantly, planning as a canonical computation could generalize not just to other navigation tasks but also to other domains, such as compositional reasoning and sequence learning, where replay has recently been demonstrated in humans (Liu et al., 2019; Schwartenbeck et al., 2021; Liu et al., 2021). Further exploring these ideas will be an exciting avenue for future work.

Author contributions

KTJ, GH, and MM conceived of the project and developed the human experimental paradigm. KTJ performed all simulations and analyzed the data. All authors interpreted the results and wrote the paper.

Acknowledgments

We are grateful to Widloski and Foster (2022) for sharing their data with us, to Ta-Chu Kao for assistance with development of the software library used to train and analyze RL agents, and to Wei Ji Ma for suggesting the method used to estimate human thinking times. We thank Nathaniel Daw, Wei Ji Ma, Zeb Kurth-Nelson, Thomas Akam, Karen Schroeder, John Widloski, Jake Stroud, Fred Callaway, Ta-Chu Kao, Evan Russek, Marine Schimel, Ji-An Li, Jeroen Olieslagers, and Sixuan Chen for helpful feedback on the manuscript. KTJ was funded by a Gates Cambridge scholarship. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising from this submission.

References

Agrawal, M., Mattar, M. G., Cohen, J. D., and Daw, N. D. (2022). The temporal dynamics of opportunity costs: A normative account of cognitive fatigue and boredom. *Psychological Review*, 129(3):564.

Alver, S. and Precup, D. (2021). What is going on inside recurrent meta reinforcement learning agents? *arXiv preprint arXiv:2104.14644*.

Antonov, G., Gagne, C., Eldar, E., and Dayan, P. (2022). Optimism and pessimism in optimised replay. *PLOS Computational Biology*, 18(1):e1009634.

Banino, A., Balaguer, J., and Blundell, C. (2021). Pondernet: Learning to ponder. *arXiv preprint arXiv:2107.05407*.

Banino, A., Barry, C., Uria, B., Blundell, C., Lillicrap, T., Mirowski, P., Pritzel, A., Chadwick, M. J., Degris, T., Modayil, J., et al. (2018). Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433.

Bansal, A., Schwarzschild, A., Borgnia, E., Emam, Z., Huang, F., Goldblum, M., and Goldstein, T. (2022). End-to-end algorithm synthesis with recurrent networks: Logical extrapolation without overthinking. *arXiv preprint arXiv:2202.05826*.

Botvinick, M., Wang, J. X., Dabney, W., Miller, K. J., and Kurth-Nelson, Z. (2020). Deep reinforcement learning and its neuroscientific implications. *Neuron*, 107(4):603–616.

Botvinick, M. M. and Cohen, J. D. (2014). The computational and neural basis of cognitive control: charted territory and new frontiers. *Cognitive science*, 38(6):1249–1285.

Callaway, F., van Opheusden, B., Gul, S., Das, P., Krueger, P. M., Griffiths, T. L., and Lieder, F. (2022). Rational use of cognitive resources in human planning. *Nature Human Behaviour*, 6(8):1112–1125.

Carr, M. F., Jadhav, S. P., and Frank, L. M. (2011). Hippocampal replay in the awake state: a potential substrate for memory consolidation and retrieval. *Nature neuroscience*, 14(2):147–153.

Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Daw, N. D., Niv, Y., and Dayan, P. (2005). Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature neuroscience*, 8(12):1704–1711.

De Lavilléon, G., Lacroix, M. M., Rondi-Reig, L., and Benchenane, K. (2015). Explicit memory creation during sleep demonstrates a causal role of place cells in navigation. *Nature neuroscience*, 18(4):493–495.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. (2016). RL2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*.

Foster, D. J. (2017). Replay comes of age. *Annu. Rev. Neurosci*, 40(581-602):9.

Geerts, J. P., Chersi, F., Stachenfeld, K. L., and Burgess, N. (2020). A general model of hippocampal and dorsal striatal learning and decision making. *Proceedings of the National Academy of Sciences*, 117(49):31427–31437.

Gillespie, A. K., Maya, D. A. A., Denovellis, E. L., Liu, D. F., Kastner, D. B., Coulter, M. E., Roumis, D. K., Eden, U. T., and Frank, L. M. (2021). Hippocampal replay reflects specific past experiences rather than a plan for subsequent choice. *Neuron*, 109(19):3149–3163.

Gomperts, S. N., Kloosterman, F., and Wilson, M. A. (2015). VTA neurons coordinate with the hippocampal reactivation of spatial experience. *Elife*, 4:e05360.

Graves, A. (2016). Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*.

Griffiths, T. L., Lieder, F., and Goodman, N. D. (2015). Rational use of cognitive resources: Levels of analysis between the computational and the algorithmic. *Topics in cognitive science*, 7(2):217–229.

Hamrick, J. B., Ballard, A. J., Pascanu, R., Vinyals, O., Heess, N., and Battaglia, P. W. (2017). Metacontrol for adaptive imagination-based optimization. *arXiv preprint arXiv:1705.02670*.

Ho, M. K., Abel, D., Correa, C. G., Littman, M. L., Cohen, J. D., and Griffiths, T. L. (2022). People construct simplified mental representations to plan. *Nature*, 606(7912):129–136.

Innes, M., Saba, E., Fischer, K., Gandhi, D., Rudilosso, M. C., Joy, N. M., Karmali, T., Pal, A., and Shah, V. (2018). Fashionable modelling with Flux. *arXiv preprint arXiv:1811.01457*.

Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. (2016). Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*.

Jadhav, S. P., Rothschild, G., Roumis, D. K., and Frank, L. M. (2016). Coordinated excitation and inhibition of prefrontal ensembles during awake hippocampal sharp-wave ripple events. *Neuron*, 90(1):113–127.

Jai, Y. Y. and Frank, L. M. (2015). Hippocampal–cortical interaction in decision making. *Neurobiology of learning and memory*, 117:34–41.

Jiang, W.-C., Xu, S., and Dudman, J. T. (2022). Hippocampal representations of foraging trajectories depend upon spatial context. *Nature neuroscience*, 25(12):1693–1705.

Jie, T. and Abbeel, P. (2010). On a connection between importance sampling and the likelihood ratio policy gradient. *Advances in Neural Information Processing Systems*, 23.

Johnson, A. and Redish, A. D. (2007). Neural ensembles in CA3 transiently encode paths forward of the animal at a decision point. *Journal of Neuroscience*, 27(45):12176–12189.

Kao, T.-C., Sadabadi, M. S., and Hennequin, G. (2021). Optimal anticipatory control as a theory of motor preparation: A thalamo-cortical circuit model. *Neuron*, 109:1567–1581.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Krause, E. L. and Drugowitsch, J. (2022). A large majority of awake hippocampal sharp-wave ripples feature spatial trajectories with momentum. *Neuron*, 110(4):722–733.

Kurth-Nelson, Z., Economides, M., Dolan, R. J., and Dayan, P. (2016). Fast sequences of non-spatial state representations in humans. *Neuron*, 91(1):194–204.

Liu, Y., Dolan, R. J., Kurth-Nelson, Z., and Behrens, T. E. (2019). Human replay spontaneously reorganizes experience. *Cell*, 178(3):640–652.

Liu, Y., Mattar, M. G., Behrens, T. E., Daw, N. D., and Dolan, R. J. (2021). Experience replay is associated with efficient nonlocal learning. *Science*, 372(6544):eabf1357.

Mattar, M. G. and Daw, N. D. (2018). Prioritized memory access explains planning and hippocampal replay. *Nature neuroscience*, 21(11):1609–1617.

Mattar, M. G. and Lengyel, M. (2022). Planning in the brain. *Neuron*.

McNamee, D. C., Stachenfeld, K. L., Botvinick, M. M., and Gershman, S. J. (2021). Flexible modulation of sequence generation in the entorhinal–hippocampal system. *Nature neuroscience*, 24(6):851–862.

Morris, R. G. (1981). Spatial localization does not require the presence of local cues. *Learning and motivation*, 12(2):239–260.

Ólafsdóttir, H. F., Carpenter, F., and Barry, C. (2017). Task demands predict a dynamic switch in the content of awake hippocampal replay. *Neuron*, 96(4):925–935.

Pascanu, R., Li, Y., Vinyals, O., Heess, N., Buesing, L., Racanière, S., Reichert, D., Weber, T., Wierstra, D., and Battaglia, P. (2017). Learning model-based planning from scratch. *arXiv preprint arXiv:1707.06170*.

Peshkin, L. and Shelton, C. R. (2002). Learning from scarce experience. *arXiv preprint cs/0204043*.

Pfeiffer, B. E. and Foster, D. J. (2013). Hippocampal place-cell sequences depict future paths to remembered goals. *Nature*, 497(7447):74–79.

Russek, E., Acosta-Kane, D., van Opheusden, B., Mattar, M. G., and Griffiths, T. (2022). Time spent thinking in online chess reflects the value of computation. *PsyArXiv*.

Samborska, V., Butler, J. L., Walton, M. E., Behrens, T. E., and Akam, T. (2022). Complementary task representations in hippocampus and prefrontal cortex for generalizing the structure of problems. *Nature Neuroscience*, 25(10):1314–1326.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. (2020). Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609.

Schwartenbeck, P., Baram, A., Liu, Y., Mark, S., Muller, T., Dolan, R., Botvinick, M., Kurth-Nelson, Z., and Behrens, T. (2021). Generative replay for compositional visual understanding in the prefrontal-hippocampal circuit. *bioRxiv*.

Solway, A. and Botvinick, M. M. (2012). Goal-directed decision making as probabilistic inference: a computational framework and potential neural correlates. *Psychological review*, 119(1):120.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

van de Ven, G. M., Trouche, S., McNamara, C. G., Allen, K., and Dupret, D. (2016). Hippocampal offline reactivation consolidates recently formed cell assembly patterns during sharp wave-ripples. *Neuron*, 92(5):968–974.

van Opheusden, B., Galbiati, G., Kuperwajs, I., Bnaya, Z., Ma, W. J., et al. (2021). Revealing the impact of expertise on human planning with a two-player board game. *PsyArXiv*.

Vul, E., Goodman, N., Griffiths, T. L., and Tenenbaum, J. B. (2014). One and done? optimal decisions from very few samples. *Cognitive science*, 38(4):599–637.

Wang, J. X., Kurth-Nelson, Z., Kumaran, D., Tirumala, D., Soyer, H., Leibo, J. Z., Hassabis, D., and Botvinick, M. (2018). Prefrontal cortex as a meta-reinforcement learning system. *Nature neuroscience*, 21(6):860–868.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. (2016). Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*.

Whittington, J. C., Muller, T. H., Mark, S., Chen, G., Barry, C., Burgess, N., and Behrens, T. E. (2020). The Tolman-Eichenbaum machine: Unifying space and relational memory through generalization in the hippocampal formation. *Cell*, 183(5):1249–1263.

Widloski, J. and Foster, D. J. (2022). Flexible rerouting of hippocampal replay sequences around changing barriers in the absence of global place field remapping. *Neuron*, 110(9):1547–1558.

Wu, C.-T., Haggerty, D., Kemere, C., and Ji, D. (2017). Hippocampal awake replay in fear memory retrieval. *Nature neuroscience*, 20(4):571–580.

Zintgraf, L., Shiarlis, K., Igl, M., Schulze, S., Gal, Y., Hofmann, K., and Whiteson, S. (2019). VariBAD: A very good method for Bayes-adaptive deep RL via meta-learning. *arXiv preprint arXiv:1910.08348*.

Supplementary figures

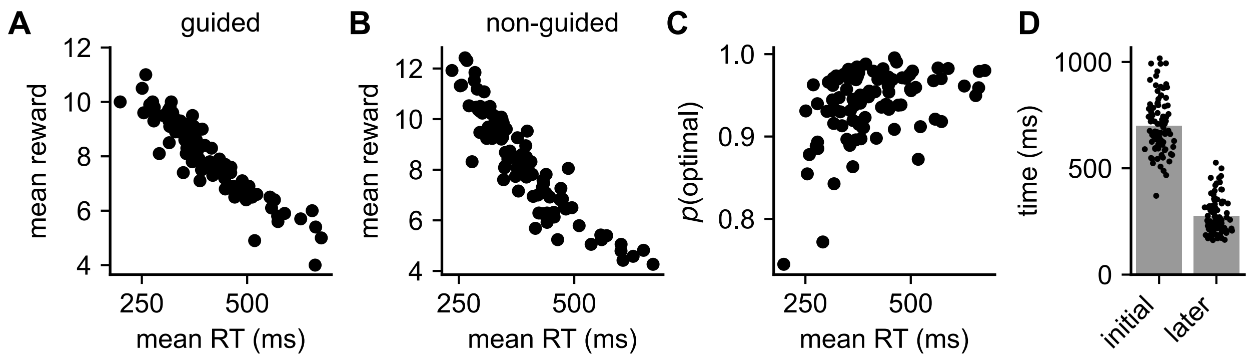

Figure S1: **Overview of human data for all participants. (A)** Mean reward per trial as a function of the average response time during the guided trials (Methods). Each data point corresponds to a single participant. **(B)** Mean reward per trial as a function of the average response time during the non-guided trials. The strong negative correlation implies that participants on average got more reward when they acted faster, confirming that participants who acted faster were not simply making random key presses. **(C)** Fraction of actions that were consistent with an optimal policy as a function of mean response time, plotted for all participants during the non-guided trials. There was a significant positive Pearson correlation between these two quantities ($r = 0.41$; $p < 0.0001$, permutation test). This correlation confirms that participants who thought for longer were not simply disengaged with the task, but that they instead invested the time to make higher-quality decisions. **(D)** Mean of the log-normal distribution of perception-action delays fitted to data from the guided episodes for each participant (dots) using either the first action within each trial (left) or all other actions (right). These prior distributions were used to infer the thinking times in Figure 2.

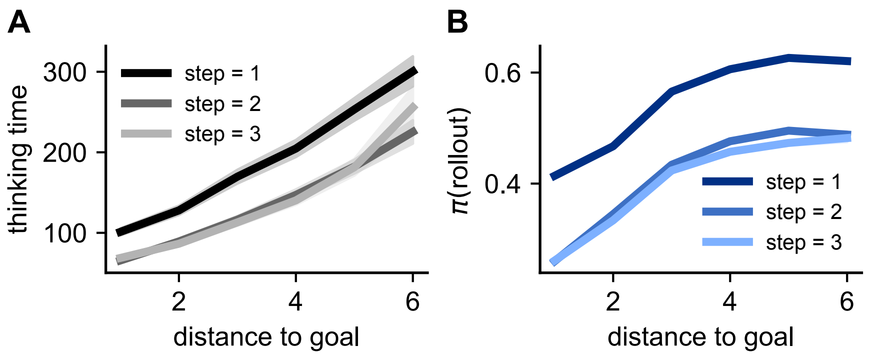

Figure S2: **Thinking time and π(rollout) by distance to goal and step within trial. (A)** Figure illustrating the average thinking time across human participants as a function of distance to goal (x-axis), conditioned on different steps within the trial (lines, legend). Subjects generally spent longer thinking before the first action of each trial, after controlling for the distance to goal, while subsequent actions were associated with similar thinking times. Lines and shadings indicate mean and standard error when repeating the analysis across human participants ($n = 94$). **(B)** π(rollout) for the agent clamped to the human trajectory as a function of distance to goal and for different steps within the trial. Similar to the human participants, the agent had a higher probability of performing a rollout on the first step of each trial. Subsequent steps were associated with similar rollout probabilities after controlling for the distance to goal. When conditioning on both distance to goal and step within trial, the residual correlation between π(rollout) and thinking time remained at a significantly positive value of $r = 0.026 \pm 0.004$ (mean $\pm$ sem).

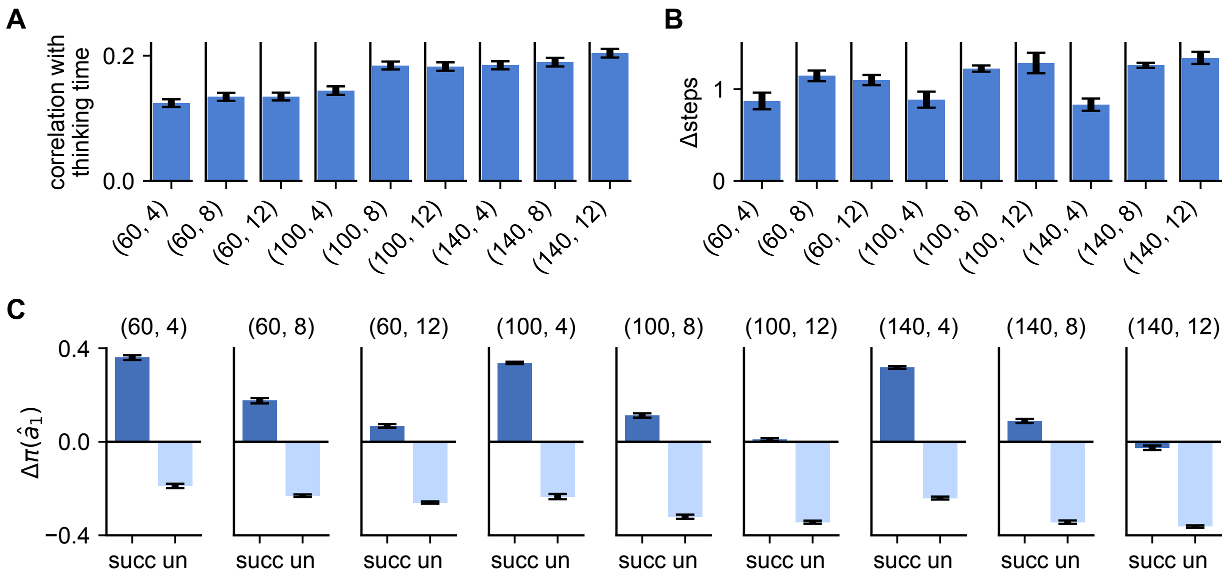

Figure S3: **Properties of networks with different hyperparameters.** To investigate the robustness of our results to the choice of network size (N) and planning horizon (L), we trained networks with each combination of $N \in \{60, 100, 140\}$ and $L \in \{4, 8, 12\}$ and repeated some of our key analyses. For all analyses, we report mean and standard error across 5 networks with each set of hyperparameters. The results in the main text are all reported for a network with $N = 100$ and $L = 8$. **(A)** We quantified the correlation between the network π(rollout) and human response times across different networks (c.f. Figure 2F). x-ticks indicate network size and planning horizon as (N, L). **(B)** We computed the improvement in the network policy from performing 5 rollouts compared to the policy in the absence of rollouts (c.f. Figure 3A). The policy improvement was quantified as the average number of steps needed to reach the goal on trial 2 in the absence of rollouts, minus the average number of steps needed with 5 rollouts enforced at the beginning of the trial and no rollouts during the rest of the trial. Positive values indicate that rollouts improved the policy. **(C)** We investigated how rollouts changed the policy (c.f. Figure 3E). For each network, we computed the average change in $\pi(\hat{a}_1)$ from before a rollout to after a rollout and report this change separately for successful ('succ') and unsuccessful ('un') rollouts. Positive values indicate that $\hat{a}_1$ became *more* likely and negative values that $\hat{a}_1$ became *less* likely after the rollout. We observe that networks with longer planning horizons tend to have less positive $\Delta\pi(\hat{a}_1)$ for successful rollouts and more negative $\Delta\pi(\hat{a}_1)$ for unsuccessful rollouts. This is consistent with a policy gradient-like algorithm with a baseline that approximates the probability of success, which increases with planning horizon. In other words, since longer rollouts are more likely to reach the goal, we should expect them to be successful and not strongly update our policy when it occurs. On the contrary, an unsuccessful rollout is less likely and should lead to a large policy change. The converse is true for shorter planning horizons.

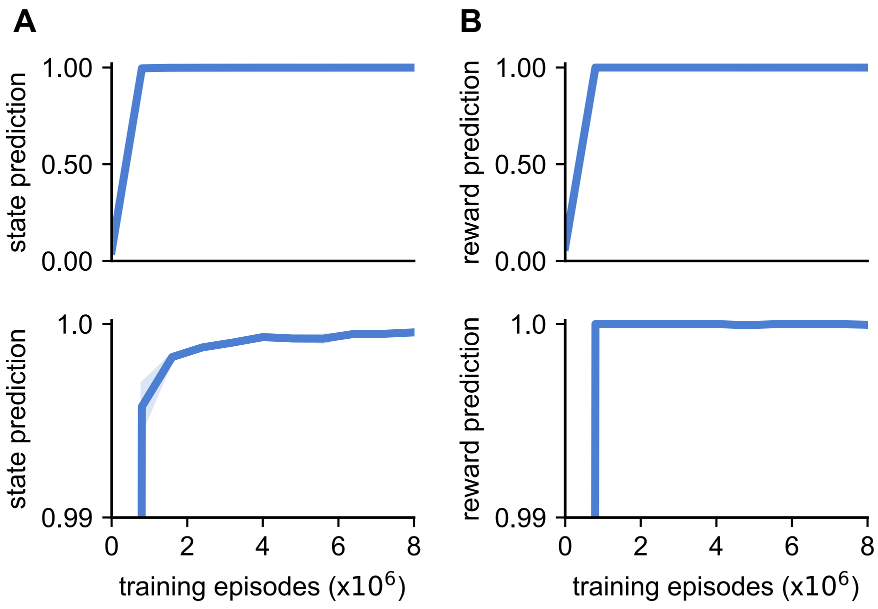

Figure S4: **Accuracy of the internal world model. (A)** Accuracy of the internal transition model over the course of training. Accuracy was computed as the probability that the predicted next state was the true state reached by the agent, ignoring all teleportation steps where the transition cannot be predicted. The accuracy was averaged across all timesteps from 1,000 episodes, and the line and shading indicate mean and standard error across 5 RL agents. The upper panel considers the full range of [0, 1] while the lower panel considers the range [0.99, 1.0]. We see that the transition model rapidly approaches ceiling performance, although it continues to improve slightly throughout training. **(B)** Accuracy of the internal reward model over the course of training. Accuracy was computed as the probability that the predicted reward location was the true reward location during the exploitation phase of the task (see Figure S5 for an analysis of the model accuracy during exploration). Lines and shadings indicate mean and standard error across 5 RL agents.

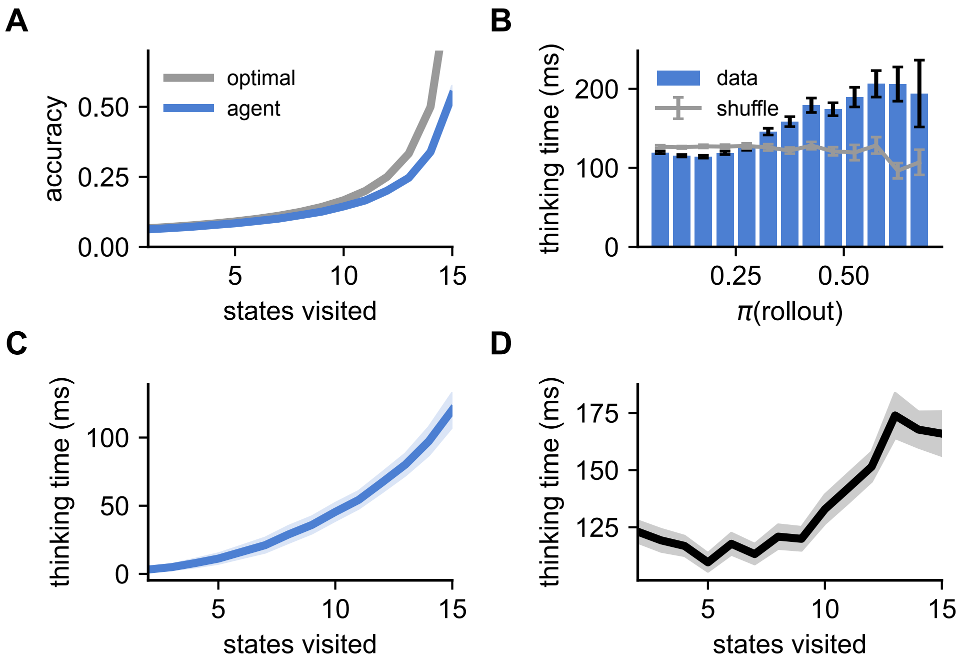

Figure S5: **Analyses of the exploration period in humans and RL agents. (A)** At each point in time, the agent outputs its belief over where the goal is located under its internal model, which was trained using a cross-entropy loss (Methods). The figure shows the average probability assigned to the true goal, plotted as a function of the number of unique states visited during the exploration phase of the task. As more states are explored, the posterior over possible goals becomes narrower and prediction accuracy increases. When the model chooses to perform a rollout, the imagined goal is chosen as the maximum likelihood location from this posterior to predict the 'success' of the rollout. The figure illustrates that this imagined goal becomes increasingly likely to be the true goal as the agent explores more of the environment. **(B)** Thinking time of human participants during exploration, plotted as a function of π(rollout) for RL agents clamped to the human trajectory. Bars and error bars indicate mean and standard error of the human thinking time across all states where π(rollout) fell in the corresponding bin. Gray line indicates a control where human thinking times have been shuffled. The Pearson correlation between π(rollout) and human thinking times is $r = 0.097 \pm 0.008$, suggesting that the model captures some of the structure in human thinking during exploration and not just during the exploitation phase. Note that the very first action of the episode was not included in this or subsequent analyses of the human data. **(C)** Model thinking time as a function of the number of unique states visited during the exploration phase of the task, with each rollout assumed to take 120 ms as specified in the main text and Methods. Line and shading indicate mean and standard error across RL agents. The increase in thinking time with visited states mirrors the predictive performance from panel (A) and suggests that the agent increasingly chooses to engage in 'model-based' planning as its uncertainty over possible goal locations decreases. **(D)** Human thinking time as a function of the number of unique states visited during the exploration phase of the task. Line and shading indicate mean and standard error across participants. The increase in thinking time with states visited suggests that humans may also transition to more model based behavior as the posterior over possible goal locations becomes narrower. A notable difference from the computational model is found early in the exploration phase, where human thinking times tend to decrease slightly over the first few unique state visits.

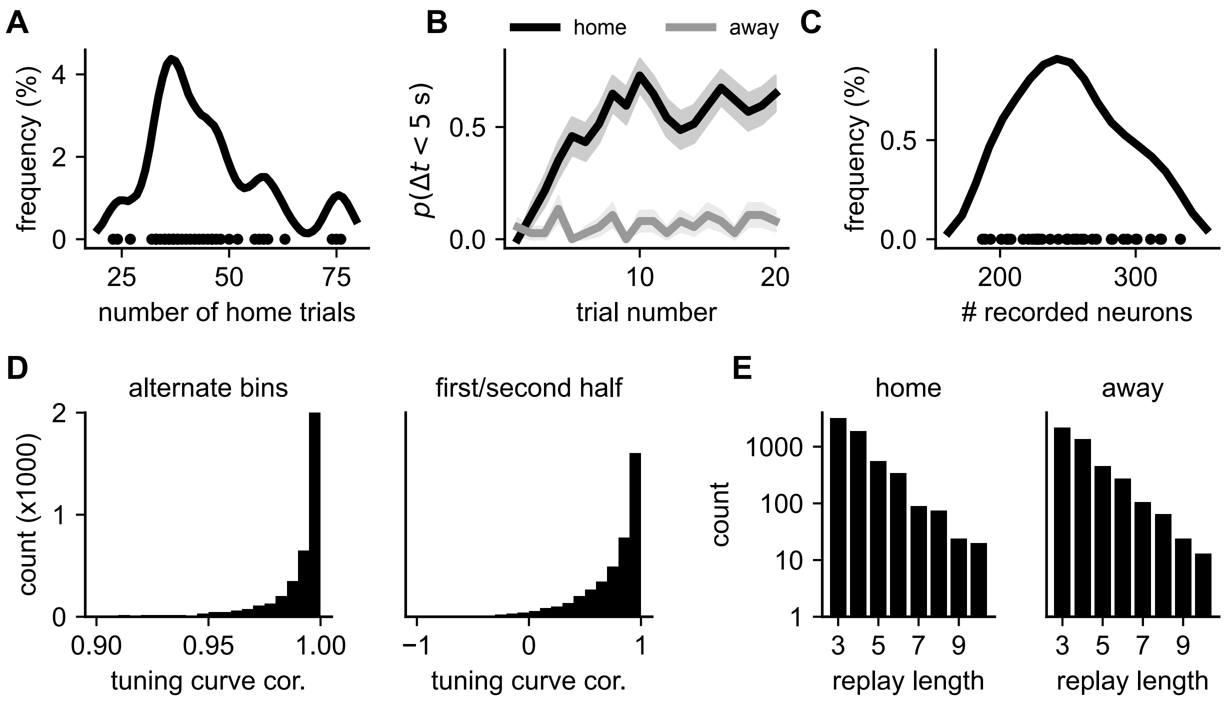

Figure S6: **Overview of rodent data.** **(A)** Kernel density estimate ($\sigma = 3$ trials) of the distribution of the number of 'home' trials in each session across all animals (an equivalent number of away trials was performed between the home trials). Dots indicate individual sessions. **(B)** Fraction of trials where the animal reached the correct goal location and started licking within 5 seconds of the trial starting, separated by home and away trials. Reaching the goal within 5 seconds was used as a success criterion by Widloski and Foster (2022) since the goal is never explicitly cued at this time (Methods). Line and shading indicate mean and standard error across sessions. The animals learn the location of the home well within a few trials and consistently return to this location on the home trials. **(C)** Distribution of the number of recorded neurons in each session. Line indicates a convolution with a Gaussian filter (15 neuron std) and dots indicate individual sessions. Note that consecutive sessions on the same day (2-3 sessions per day) involved recording from the same neurons, so there are fewer distinct data points than there are sessions. **(D)** Consistency of spatial tuning curves of hippocampal neurons. Consistency was quantified by constructing two tuning curves on the 5x5 spatial grid (Figure 4A) for each neuron and computing the Pearson correlation between the two tuning curves. The data was split into either even/odd time bins in a session (left plot) or first/second half of the session (right plot) to compute a pair of tuning curves. **(E)** Distribution of replay lengths, measured as the number of states visited in a replay, for all replays during home (left) or away (right) trials. Note the log scale on the y-axis.

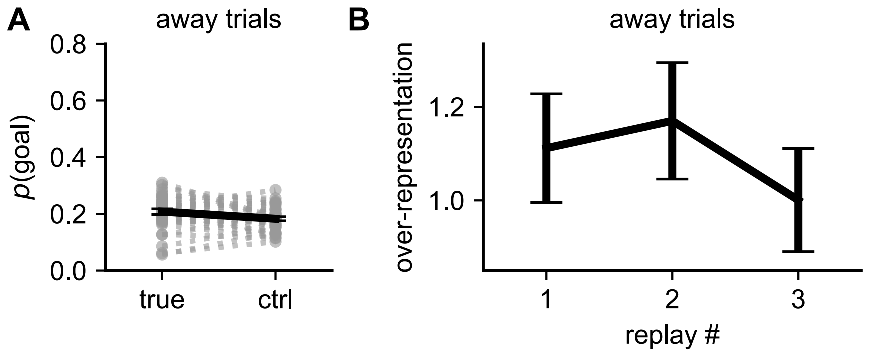

Figure S7: **Analysis of replays during away trials. (A)** Fraction of replays reaching either the true goal (left) or a randomly sampled alternative goal location (right) during away trials. In contrast to the home trials (Figure 4C), the goal is not over-represented during away trials, where the goal location is unknown. **(B)** Over-representation of replay success as a function of replay number within sequences of replays containing at least 3 distinct replay events (c.f. Figure 4E). In contrast to the home trials, there is no increase in over-representation with replay number during these away trials.

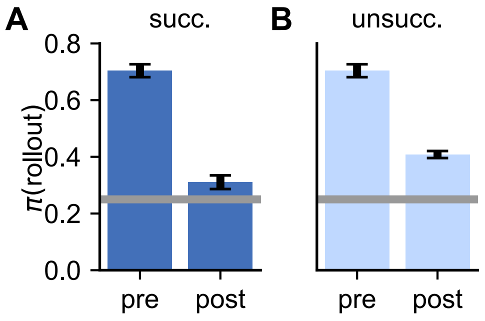

Figure S8: **Change in π(rollout) for successful and unsuccessful rollouts. (A)** π(rollout) before (left) and after (right) successful rollouts. Bars and error bars indicate mean and standard error across 5 RL agents. The data used for this analysis was the same data used in Figure 3E. **(B)** As in (A), now for unsuccessful rollouts. π^{post}(rollout) was substantially larger after unsuccessful than successful rollouts ($\Delta\pi(\text{rollout}) = 0.10 \pm 0.01$ mean $\pm$ sem).

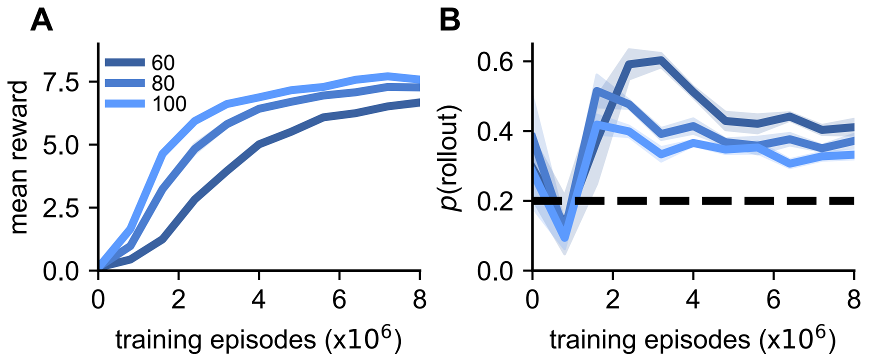

Figure S9: **Performance and rollouts as a function of network size. (A)** We trained networks of different sizes (legend; $N \in [60, 80, 100]$) and quantified their performance over the course of training. **(B)** Fraction of timesteps where the agent chose to perform a rollout over the course of training for different network sizes. Note that the agents perform rollouts at chance level but with high variance at initialization, and this data point was therefore not included in the analysis in Figure 5E, where we only considered the learned rollout frequency from episode 800,000 onwards. It is interesting to note that the agents first learn to *suppress* the rollout frequency below chance before increasing it to levels above chance. This is consistent with a theory where rollouts only become useful when (i) an internal world model has been learned, and (ii) the agent has learned *how* to use rollouts to improve its policy. Finally, rollouts become less frequent again later in training as the base policy improves.

Methods

Software

All models were trained in Julia version 1.7 using Flux and Zygote for automatic differentiation (Innes et al., 2018). Human behavioral experiments were written in OCaml, with the front-end transpiled to javascript for running in the participants' browsers. All analyses of the models and human data were performed in Julia version 1.8. All analyses of hippocampal replay data were performed in Python 3.8.

Statistics

Unless otherwise stated, all plots are reported as mean and standard error across human participants ($n = 94$), independently trained RL agents ($n = 5$), or experimental sessions in rodents ($n = 37$).

Environment

We generated mazes using the following algorithm:

Algorithm 1: Maze generating algorithm

```
1  $\mathcal{A} \leftarrow$ 4x4 arena with walls everywhere.
2  $\mathcal{V} \leftarrow \{\}$ % empty initial set of visited states.
3  $s \leftarrow$ random starting location.
4
5  % Define function to walk through the maze and remove walls
6  Function walk_maze($s$, $\mathcal{A}$, $\mathcal{V}$)
7      $\mathcal{V}$.add($s$) % Add $s$ to set of visited states
8      $\mathcal{N} \leftarrow$ neighbors($s$) % Neighbors of $s$, including those through the periodic boundaries
9      % Iterate through all neighboring states in random order
10     for $n \in$ randomize($\mathcal{N}$) do
11         % If we reached a state we have not seen before
12         if $n \notin \mathcal{V}$ then
13             $\mathcal{A}$.remove_wall($s$, $n$) % Remove wall between $s$ and $n$ from arena
14             $\mathcal{A}$, $\mathcal{V}$ = walk_maze($n$, $\mathcal{A}$, $\mathcal{V}$) % Continue from new state
15     return $\mathcal{A}$, $\mathcal{V}$
16
17 $\mathcal{A}$, $\mathcal{V}$ = walk_maze($s$, $\mathcal{A}$, $\mathcal{V}$) % Construct maze using our recursive algorithm
18
19 %Remove 3 additional walls at random to increase the degeneracy of the tasks.
20 %This increases the number of decision points with multiple routes to the goal.
21 for $i$ = 1:3 do
22     $w$ = random_wall($\mathcal{A}$) % Select one of the remaining walls at random
23     $\mathcal{A}$.remove_wall($w$) % Remove from set of walls
24
25 return $\mathcal{A}$ % Return the maze we constructed
```

For each environment, a goal location was sampled uniformly at random. When subjects took an action leading to the goal, they transitioned to this location before being teleported to a random location. In the computational model, this was achieved by feeding the agent an input at this location before teleporting the agent to the new location. The policy of the agent at this iteration of the network dynamics was ignored, since the agent was teleported rather than taking an action.

Reinforcement learning model

We trained our agent to maximize the expected reward, with the expectation taken both over environments $\mathcal{E}$ and the agent's policy π:

$$\mathcal{U} = \mathbb{E}_{\mathcal{E}}\left[J(\theta)\right] \tag{5}$$

$$= \mathbb{E}_{\mathcal{E}}\left[\mathbb{E}_{\pi}\left(\sum_{k=1}^{K} r_k\right)\right]. \tag{6}$$

Here, $\mathcal{U}$ is the *utility function*, k indicates the iteration within an episode, and r_k indicates the instantaneous reward at each iteration. We additionally introduced the following auxiliary losses:

$$\mathcal{L}_V = 0.5(V_k - R_k)^2 \qquad \text{value function} \tag{7}$$

$$\mathcal{L}_H = \mathbb{E}_{\pi} \log \pi \qquad \text{entropy regularization} \tag{8}$$

$$\mathcal{L}_P = -\sum_i \left[s_{k+1}^{(i)} \log \hat{s}_{k+1}^{(i)} + g^{(i)} \log \hat{g}_k^{(i)}\right] \qquad \text{internal world model.} \tag{9}$$

Here, $\hat{\boldsymbol{g}}_k$, and $\hat{\boldsymbol{s}}_{k+1}$ are additional network outputs representing the agent's estimate of the current reward location and upcoming state. $\boldsymbol{g}$ and $\boldsymbol{s}_{k+1}$ are the corresponding ground truth quantities, represented as one-hot vectors. $R_k := \sum_{k'=k}^{K} r_{k'}$ is the empirical cumulative future reward from iteration k onwards, and V_k is the value function of the agent.

To maximize the utility and minimize the losses, we trained the RL agent on-policy using a policy gradient algorithm with a baseline (Sutton and Barto, 2018) and parameter updates of the form

$$\Delta\theta \propto \sum_{a_k \sim \pi}\left[(\underbrace{\nabla_\theta \log \pi_k(a_k)}_{\text{actor}} + \underbrace{\beta_v \nabla_\theta V_k}_{\text{critic}})\delta_k - \beta_e \nabla_\theta \sum_a \underbrace{\pi_{k,a} \log \pi_{k,a}}_{\text{entropy}} + \underbrace{\beta_p \Delta\theta_p}_{\text{predictive}}\right] \tag{10}$$

Here, $\delta_k := -V_k + R_k$ is the 'advantage function', and $\Delta\theta_p = \nabla_\theta \mathcal{L}_P$ is the derivative of the predictive loss $\mathcal{L}_P$, which was used to train the 'internal model' of the agent. $\beta_p = 0.5$, $\beta_v = 0.05$ and $\beta_e = 0.05$ are hyperparameters controlling the importance of the three auxiliary losses. While we use the predictive model explicitly in the planning loop, similar auxiliary losses are also commonly used to speed up training by encouraging the learning of useful representations (Jaderberg et al., 2016).

Our model consisted of a GRU network with 100 hidden units (Cho et al., 2014). The policy was computed as a linear function of the hidden state followed by a softmax normalization. The value function was computed as a linear function of the hidden state. The predictions of the next state and reward location were computed with a neural network that received as input a concatenation of the current hidden state $\boldsymbol{h}_k$ and the action a_k sampled from the policy (as a one-hot representation). The output layer of this feedforward network was split into a part that encoded a distribution over the predicted next state (a vector of 16 grid locations with softmax normalization), and a part that encoded the predicted reward location in the same way. This network had a single hidden layer with 33 units and a ReLU nonlinearity.

The model was trained using ADAM (Kingma and Ba, 2015) on 200,000 batches, each consisting of 40 episodes, for a total of 8×10^6 training episodes. These episodes were sampled independently from a total task space of $(273 \pm 13) \times 10^6$ tasks (mean $\pm$ standard error). The total task space was estimated by sampling 50,000 wall configurations and computing the fraction of the resulting 1.25×10^9 pairwise comparisons that were identical, divided by 16 to account for the possible reward locations. This process was repeated 10 times to estimate a mean and confidence interval. These considerations suggest that the task coverage during training was $\approx 2.9\%$, which confirms that the majority of tasks seen at test time are novel (although we do not enforce this explicitly).

For all evaluations of the model, actions were sampled greedily rather than on-policy unless otherwise stated. This was done since the primary motivation for using a stochastic policy is to explore the space of policies to improve learning, and performance was better under the greedy policy at test time.

Planning

Our implementation of 'planning' in the form of policy rollouts is described in Algorithm 2. This routine was invoked whenever a 'rollout' was sampled from the policy instead of a physical action.

Algorithm 2: Planning routine for the RL agent

```
1  input: maximum planning depth ($n_{max}$), current hidden state ($\boldsymbol{h}_k$), and agent location ($\boldsymbol{s}_k$)
2  parameters: network parameters $\theta$, defining $\phi(\cdot)$, $\zeta(\cdot)$, $p(\hat{\boldsymbol{g}}|\boldsymbol{h}_k)$, and $p(\hat{\boldsymbol{s}}|a, \boldsymbol{h})$
3
4  $\tilde{\boldsymbol{g}} \leftarrow \operatorname{argmax} p(\hat{\boldsymbol{g}}|\boldsymbol{h}_k)$ % predicted goal location
5  $\tilde{\boldsymbol{h}}_k, \tilde{\pi}_k, \tilde{\boldsymbol{s}}_k \leftarrow \boldsymbol{h}_k, \pi_k, \boldsymbol{s}_k$ % simulated hidden state, policy, and agent location, initialized to true values
6  $n \leftarrow 0$ % planning iteration
7
8  while $n < n_{max}$ and $\tilde{\boldsymbol{s}}_{k+n} \neq \tilde{\boldsymbol{g}}$ do
9      $\tilde{a}_{k+n} \sim \tilde{\pi}_{k+n}[\{a\}_{\text{no\_plan}}]$ % imagined action sampled on-policy but from physical actions only
10     $\tilde{\boldsymbol{s}}_{k+n+1} \leftarrow \operatorname{argmax} p(\hat{\boldsymbol{s}}_{k+n+1}|\tilde{a}_{k+n}, \tilde{\boldsymbol{h}}_{k+n})$ % predicted next state from current imagined state and action
11     $\tilde{\boldsymbol{x}}_{k+n+1} \leftarrow O(\tilde{\boldsymbol{s}}_{k+n+1}, \tilde{\boldsymbol{g}})$ % expected observations on next iteration (assuming access to the function $O(\cdot)$)
12     $\tilde{\boldsymbol{h}}_{k+n+1} \leftarrow \phi(\tilde{\boldsymbol{x}}_{k+n+1}, \tilde{\boldsymbol{h}}_{k+n})$ % simulate agent dynamics
13     $\tilde{\pi}_{k+n+1} = \zeta(\tilde{\boldsymbol{h}}_{k+n+1})$ % generate new policy
14     $n \leftarrow n + 1$ % update planning iteration
15
16 % return action sequence and whether the rollout reached the expected goal
17 return: $\{\tilde{a}_{k'}\}_k^{k+n}$, $\delta(\tilde{\boldsymbol{s}}_{k+n}, \tilde{\boldsymbol{g}})$
```

For the network update following a rollout, the input $\boldsymbol{x}_{k+1}$ was augmented with an additional 'rollout input' consisting of (i) the sequence of simulated actions, each as a 1-hot vector, and (ii) a binary input indicating whether the imagined sequence of states reached the imagined goal location. Additionally, the time within the session was only updated by 120 ms after a rollout in contrast to the 400 ms update after a physical action or teleportation step.

Note that while both an imagined 'physical state' $\tilde{\boldsymbol{s}}_k$ and 'hidden state' $\tilde{\boldsymbol{h}}_k$ are updated during the rollout, the agent continues from the *original* location $\boldsymbol{s}_k$ and hidden state $\boldsymbol{h}_k$ after the rollout, but with an augmented input. Additionally, gradients were not propagated through the rollout process, which was considered part of the 'environment'. This means that there was no explicit gradient signal that encouraged the policy to drive useful or informative rollouts. Instead, the rollout process simply relied on the utility of the base policy optimized for acting in the environment.

Performance by number of rollouts

To quantify the performance as a function of the number of planning steps in the RL agent (Figure 3A), we simulated each agent in 1,000 different mazes until it first found the goal and was teleported to a random location. We then proceeded to enforce a particular number of rollouts before the agent was released in trial 2. During this release phase, no more rollouts were allowed – in other words, the policy was re-normalized over the physical actions, and the probability of performing a rollout was set to zero. Performance was then quantified as the average number of steps needed to reach the goal during this test phase. The optimal reference value was computed as the average optimal path length for the corresponding starting states. When performing more than one sequential rollout prior to taking an action, the policy of the agent can continue to change through two potential mechanisms. The first is that the agent can explicitly 'remember' the action sequences from multiple rollouts and somehow arbitrate between them. The second is to progressively update the hidden state in a way that leads to a better expected policy with each rollout, since the feedback from a rollout is incorporated into the hidden state that induces the policy used to draw the next rollout. On the basis of the analysis in Figure 5, we expect the second mechanism to be dominant, although we did not explicitly test the ability of the agent to ‘remember’ multiple action sequences from sequential rollouts. For this and all other RNN analyses, the agent executed the most likely action under the policy during ‘testing’ in contrast to the sampling performed during training, where such stochasticity is necessary for exploring the space of possible actions. All results were qualitatively similar if actions were sampled during the test phase, although average performance was slightly worse.

Performance in the absence of rollouts and with shuffled rollout times

To quantify the performance of the RL agent in the absence of rollouts, we let the agent receive inputs and produce outputs as normal. However, we set the probability of performing a rollout under the policy to zero and re-normalized the policy over the physical actions before choosing an action from the policy. We compared the average performance of the agent (number of rewards collected) in this setting to the performance of the default agent in the same environments.

To compare the original performance to an agent with randomized rollout times, we counted the number of rollouts performed by the default agent in each environment. We then re-sampled a new set of network iterations at which to perform rollouts, matching the size of this new set to the original number of rollouts performed in the corresponding environment. Finally, we let the agent interact with the environment again, while enforcing a rollout on these network iterations, and preventing rollouts at all other timesteps. It is worth noting that we could not predict *a priori* the iterations on which the agent would find the goal, at which point rollouts were not possible. If a rollout had been sampled at such an iteration, we re-sampled this rollout from the set of remaining network iterations.

Rollouts by network size

To investigate how the frequency of rollouts depended on network size (Figure 5E; Figure S9), we trained networks with either 60, 80, or 100 hidden units (GRUs). Five networks were trained with each size. At regular intervals during training, we tested the networks on a series of 5,000 mazes and computed (i) the average reward per episode, and (ii) the fraction of actions that were rollouts rather than physical actions. We then plotted the rollout fraction as a function of average reward to see how frequently an agent of a given size performed rollouts for a particular performance.

Effect of rollouts on agent policy

To quantify the effect of rollouts on the policy of the agent, we simulated each agent in 1,000 different mazes until it first found the goal and was teleported to a random location. We then resampled rollouts until both (i) a successful rollout and (ii) an unsuccessful rollout had been sampled. Finally, we quantified $\pi^{pre}(\hat{a}_1)$ and $\pi^{post}(\hat{a}_1)$ separately for the two scenarios and plotted the results in Figure 3E. Importantly, this means that each data point in the ‘successful’ analysis had a corresponding data point in the ‘unsuccessful’ analysis with the exact same maze, location, and hidden state. In this way, we could query the effect of rollouts on the policy without the confound of how the policy itself affects the rollouts. For this analysis, we discarded episodes where the first 100 sampled rollouts did not result in both a successful and an unsuccessful rollout.

For Figure S8, we used the same episodes and instead quantified $\pi(\text{rollout})$ before and after the rollout, repeating the analysis for both successful and unsuscсessful rollouts.

Overlap between hidden state updates and policy gradients

Using a single rollout ($\hat{\tau}$) to approximate the expectation over trajectories of the gradient of the expected future reward for a given episode, $\nabla_{\boldsymbol{h}} J_{fut}(\boldsymbol{h})$, the policy gradient update in $\boldsymbol{h}$ takes the form $\Delta\boldsymbol{h} \propto (R_{\hat{\tau}} - b)\nabla_{\boldsymbol{h}} \log p(\hat{\tau})$. Here, $\Delta\boldsymbol{h}$ is the change in hidden state resulting from the rollout, $R_{\hat{\tau}}$ is the ‘reward’ of the simulated trajectory, b is a constant or state-dependent baseline, and $\nabla_{\boldsymbol{h}} \log p(\hat{\tau})$ is the gradient with respect to the hidden state of the log probability of $\hat{\tau}$ under the policy induced by $\boldsymbol{h}$. This implies that the *derivative* of the hidden state update w.r.t. $R_{\hat{\tau}}$, $\boldsymbol{\alpha}^{\text{RNN}} := \frac{\partial \Delta\boldsymbol{h}}{\partial R_{\hat{\tau}}}$, should be proportional to $\boldsymbol{\alpha}^{\text{PG}} := \nabla_{\boldsymbol{h}} \log p(\hat{\tau})$.

For these analyses, we divided $\hat{\tau}$ into its constituent actions, defining $\boldsymbol{\alpha}^{\text{PG}}_k := \nabla_{\boldsymbol{h}} \log p(\hat{a}_k | \hat{a}_{1:k-1})$ as the derivative w.r.t. the hidden state of the log probability of taking the simulated action at step k, conditioned on the actions at all preceding steps (1 to $k-1$) being consistent with the rollout. To compute $\boldsymbol{\alpha}^{\text{RNN}}$, we also needed to take derivatives w.r.t. $R_{\hat{\tau}}$ – the ‘reward’ of a rollout. A naive choice here would be to simply consider $R_{\hat{\tau}}$ to be the input specifying whether the rollout reached the reward or not. However, we hypothesized that the agent would also use information about e.g. how long the simulated trajectory was in its estimate of the 'goodness' of a rollout (since a shorter rollout implies that the goal was found faster). We therefore determined the direction in planning input state space that was most predictive of the time-to-goal of the agent. We did this by using linear regression to predict the (negative) time-to-next-reward as a function of the planning feedback $\boldsymbol{x}_f$ across episodes and rollouts. This defines the (normalized) direction $\hat{\boldsymbol{\nu}}$ in planning input space that maximally increases the expected future reward. Finally, we defined $R_{\hat{\tau}}$ as the magnitude of the planning input in direction $\hat{\boldsymbol{\nu}}$, $R_{\hat{\tau}} := \boldsymbol{x}_f \cdot \hat{\boldsymbol{\nu}}$. We could then compute $\boldsymbol{\alpha}^{\text{RNN}}$ with this definition of $R_{\hat{\tau}}$ using automatic differentiation.

In Figure 5C, we computed $\boldsymbol{\alpha}^{\text{RNN}}$ and $\boldsymbol{\alpha}_1^{\text{PG}}$ across 1,000 episodes. We then performed PCA on the set of $\boldsymbol{\alpha}_1^{\text{PG}}$ and projected both $\boldsymbol{\alpha}^{\text{RNN}}$ and $\boldsymbol{\alpha}_1^{\text{PG}}$ into the space spanned by the top 3 PCs. Finally, we computed the mean value of both quantities conditioned on $\hat{a}_1$ to visualize the alignment. In Figure 5D, we considered the same $\boldsymbol{\alpha}^{\text{RNN}}$ and $\boldsymbol{\alpha}_1^{\text{PG}}$ vectors, now computing the cosine similarity between each pair of vectors before taking an average. This cosine similarity was still computed in the space spanned by the top 3 PCs since we were primarily interested in changes in $\boldsymbol{h}$ within the subspace that would affect $\log p(\hat{\tau})$. As a control, we repeated the analysis after altering the planning input $\boldsymbol{x}_f$ to falsely inform the agent that it had simulated some other action $\hat{a}_{1,\text{ctrl}} \neq \hat{a}_1$. Finally, we also repeated this analysis using $\boldsymbol{\alpha}_2^{\text{PG}}$ to characterize how the effects of the planning input propagated through the recurrent network dynamics to modulate future action probabilities.

Human data collection

The human behavioral experiment used in this study has been certified as exempt from IRB review by the UC San Diego Human Research Protection Program. We collected data from 100 human participants (50 male, 50 female) recruited on Prolific to perform the task described in Figure 1B. Subjects were asked to complete (i) 6 'guided' episodes where the optimal path was shown explicitly, followed by (ii) 40 non-guided episodes, and (iii) 12 guided episodes. The task can be found online. During data collection, a subject was deemed 'disengaged', and the trial repeated, if one of three conditions were met: (i) the same key was pressed 5 times in a row, (ii) the same key pair was pressed four times in a row, or (iii) no key was pressed for 7 seconds. Participants were paid a fixed rate of \$3 plus a performance-dependent bonus of \$0.002 for each completed trial across both guided and non-guided episodes. The experiment took approximately 22 minutes to complete, and the average pay across participants was \$10.5 per hour including the performance bonus.

The data from 6 participants with a mean response time greater than > 690 ms during the guided episodes were excluded to avoid including participants who were not sufficiently engaged with the task. For the guided episodes, only the last 10 episodes were used for further analyses. For the non-guided episodes, we discarded the first two episodes and used the last 38 episodes. This was done to give participants two episodes to get used to the task for each of the two conditions, and the first set of guided episodes was intended as an instruction in how to perform the task.

Performance as a function of trial number

We considered all episodes where the humans or RL agents completed at least four trials, evaluating the RL agents across 50,000 episodes. We then computed the average across these episodes of the number of steps to goal as a function of trial number separately for all subjects. Figure 2A illustrates the mean and standard error across subjects (human participants or RL agents). The optimal value during the exploitation phase was computed by using dynamic programming to find the shortest path between each possible starting location and the goal location, averaged across all environments seen by the RL agent. To compute the exploration baseline, brute force search was used to identify the path that explored the full environment as fast as possible. The optimal exploration performance was then computed as the expected time-to-first-reward under this policy, averaged over all possible goal locations.

Estimation of thinking times

In broad strokes, we assumed that for each action, the response time t_r is the sum of a thinking time t_t and some perception-action delay, both subject to independent variability:

$$t_\text{r} = t_\text{t} + t_\text{d} \quad \text{with} \quad t_\text{t} \sim p_\text{t} \quad \text{and} \quad t_\text{d} \sim p_\text{d}. \tag{11}$$

Here, $\{t_\mathrm{r}, t_\mathrm{t}, t_\mathrm{d}\} \geq 0$ since elapsed time cannot be negative. We assumed that the prior distribution over perception-action delays, p_d, was identical during guided and non-guided trials. For each subject, we obtained a good model of p_d (see below) by considering the distribution of response times measured during guided trials. This was possible because guided trials involved no 'thinking' by definition, such that $t_\mathrm{d} \equiv t_\mathrm{r}$ was directly observed. Finally, for any non-guided trial with observed response t_r, we formed a point estimate of the thinking time by computing the mean of the posterior $p(t_\mathrm{t}|t_\mathrm{r})$:

$$\hat{t}_{\mathrm{t}|t_\mathrm{r}} = \mathbb{E}_{p(t_\mathrm{t}|t_\mathrm{r})}[t_\mathrm{t}]. \tag{12}$$

In more detail, we took p_t during guided trials to be uniform between 0 and 7 s – the maximum response time allowed, beyond which subjects were considered disengaged, and the trial was discarded and reset. For $p_\mathrm{d}(t_\mathrm{d})$, we assumed a shifted log-normal distribution,

$$p_\mathrm{d}(t_\mathrm{d}; \mu, \sigma, \delta) = \begin{cases} \frac{1}{(t_\mathrm{d}-\delta)\sigma\sqrt{2\pi}} \exp\left[-\frac{(\log(t_\mathrm{d}-\delta)-\mu)^2}{2\sigma^2}\right] & \text{if } t_d > \delta \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

with parameters μ, σ, and δ obtained via maximum likelihood estimation based on on the collection of response times $t_\mathrm{r} \equiv t_\mathrm{d}$ observed during guided trials. For a given δ, the maximum likelihood values of μ and σ are simply given by the mean and standard deviation of the logarithm of the observations. To fit this shifted log-normal model, we thus performed a grid search over $\delta \in [0, \min(t_r^\text{guided}) - 1]$ at 1 ms resolution and selected the value under which the optimal (μ, σ) gave the largest likelihood. This range of δ was chosen to ensure that (i) only positive values of t_r^guided had positive probability, and (ii) all observed t_r^guided had non-zero probability. We then retained the optimal μ, σ, and δ to define the prior over $p_\mathrm{d}(t_d)$ on guided trials for each subject.

According to Bayes' rule, the posterior is proportional to

$$p(t_\mathrm{t}|t_\mathrm{r}) \propto p(t_\mathrm{r}|t_\mathrm{t})p(t_\mathrm{t}) \tag{14}$$

where

$$\begin{aligned} p(t_\mathrm{r}|t_\mathrm{t}) &= \int_0^\infty dt_\mathrm{d}\, p_\mathrm{d}(t_\mathrm{d})\, p(t_\mathrm{r}|t_\mathrm{t}, t_\mathrm{d}) & (15) \\ &= \int_0^\infty dt_\mathrm{d}\, p_\mathrm{d}(t_\mathrm{d})\, \delta(t_\mathrm{d} - (t_\mathrm{r} - t_\mathrm{t})) & (16) \\ &= p_\mathrm{d}(t_\mathrm{r} - t_\mathrm{t}) & (17) \end{aligned}$$

Therefore, the posterior is given by

$$p(t_\mathrm{t}|t_\mathrm{r}) \propto \begin{cases} p_\mathrm{d}(t_\mathrm{r} - t_\mathrm{t}) & \text{if } t_\mathrm{t} > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{18}$$

resulting in the following posterior mean:

$$\hat{t}_{\mathrm{t}|t_\mathrm{r}} := \mathbb{E}_{p(t_\mathrm{t}|t_\mathrm{r})}[t_\mathrm{t}] = t_\mathrm{r} - \int_\delta^{t_\mathrm{r}} t_\mathrm{d}\, p_\mathrm{d}(t_\mathrm{d}|t_\mathrm{d} < t_\mathrm{r}; \mu, \sigma, \delta)\, dt_\mathrm{d}. \tag{19}$$

Here, $p_\mathrm{d}(t_\mathrm{d}|t_\mathrm{d} < t_\mathrm{r})$ denotes $p_\mathrm{d}(t_\mathrm{d})$ re-normalized over the interval $t_\mathrm{d} < t_\mathrm{r}$, and the condition $(t_\mathrm{d} < t_\mathrm{r})$ is equivalent to $(t_\mathrm{t} > 0)$. We note that the integral runs from δ to t_r since $p_\mathrm{d}(t_\mathrm{d}) = 0$ for $t_\mathrm{d} < \delta$. As δ simply shifts the distribution over t_d, we can rewrite this as

$$\hat{t}_{\mathrm{t}|t_\mathrm{r}} = t_\mathrm{r} - \delta - \int_0^{t_\mathrm{r}-\delta} x\, p_\mathrm{d}(x|x < t_\mathrm{r} - \delta; \mu, \sigma, \delta = 0)\, dx. \tag{20}$$

This is useful since the conditional expectation of a log-normally distributed random variable with $\delta = 0$ is given in closed form by

$$\mathbb{E}_{\mu,\sigma}[x|x<k] = \int_0^k x\, p(x|x<k;\mu,\sigma,\delta=0)\, dx \tag{21}$$

$$= \exp[\mu + 0.5\sigma^2] \frac{\Phi\left(\frac{\log(k)-\mu-\sigma^2}{\sigma}\right)}{\Phi\left(\frac{\log(k)-\mu}{\sigma}\right)}, \tag{22}$$

where $\Phi(\cdot)$ is the cumulative density function of the standard Gaussian, $\mathcal{N}(0,1)$. This allows us to compute the posterior mean thinking time for an observed response time t_r in closed form as

$$\hat{t}_{t|t_\mathrm{r}} = t_\mathrm{r} - \delta - \mathbb{E}_{\mu,\sigma}[x|x<t_\mathrm{r}-\delta]. \tag{23}$$

We note that the support of $p_\mathrm{d}(t_\mathrm{d}|t_\mathrm{d} < t_\mathrm{r};\mu,\sigma,\delta)$ is $t_\mathrm{d} \in [\delta, t_\mathrm{r}]$. For 0.6% of the non-guided decisions, the value of t_r was *lower* than the estimated δ for the corresponding participant, in which case $p(t_\mathrm{t}|t_\mathrm{r})$ is undefined. In such cases, we defined the thinking time to be $\hat{t}_{\mathrm{t}|t_\mathrm{r}} = 0$, since the response time was shorter than our estimated minimum perception-action delay. A necessary (but not sufficient) condition for $t_\mathrm{r} < \delta$ is that t_r is smaller than the smallest response time in the guided trials.

The whole procedure of fitting and inference described above was repeated separately for actions that immediately followed a teleportation step (i.e. the first action in each trial) and for all other actions. This is because we expected the first action in each trial to be associated with an additional perceptual delay compared to actions that followed a predictable transition.

All results were qualitatively similar using other methods for estimating thinking time, including (i) a log-normal prior over t_d with no shift ($\delta = 0$), (ii) using the posterior mode instead of the posterior mean, (iii) estimating a constant t_d from the guided trials, and (iv) estimating a constant t_d as the 0.1 or 0.25 quantile of t_r from the non-guided trials.

Thinking times in different situations

To investigate how the thinking time varied in different situations, we considered only exploitation trials and computed for every action (i) the minimum distance to the goal *at the beginning of the corresponding trial*, and (ii) what action number this was within the trial. We then computed the mean thinking time as a function of action number separately for each distance-to-goal. This analysis was repeated across experimental subjects and results reported as mean and standard error across subjects.

We repeated this analysis for the RL agents, where 'thinking time' was now defined based on the average number of rollouts performed, conditioned on action-within-trial and original distance to goal.

Comparison of human and model thinking times

For each subject and each RL agent, we clamped the trajectory of the agent to that taken by the subject (i.e. we used the human actions instead of sampling from the policy). After taking an action, we recorded π(rollout) under the model on the first timestep of the new state for comparison with human thinking times. We then sampled a rollout with probability π(rollout) and took an action (identical to the next human action) with probability $1 - \pi$(rollout), repeating this process until the next state was reached. Finally, we computed the average π(rollout) across 20 iterations of each RL agent for comparison with the human thinking time in each state. Figure 2E shows the human thinking time as a function of π(rollout), with the bars and error bars illustrating the mean and standard error in each bin. For this analysis, data was aggregated across all participants. Results were similar if we compared human thinking times with the average number of rollouts performed rather than the initial π(rollout).

In Figure 2F, we computed the correlation between thinking time and various regressors on a participant-by-participant basis and report the result as mean and standard error across participants ($n = 94$). For the 'residual' correlation, we first computed the mean thinking time for each distance-to-goal for each participant and the corresponding mean π(rollout) for the RL agents. We then subtracted the appropriate mean values from the thinking times and π(rollout) in the human participants and RL agents. In other words, we subtracted the average thinking time for situations 5 steps from the goal from all data points where the participant was 5 steps from the goal etc. Finally, we computed the correlation between the residual π(rollout) and the residual thinking times. This analysis was repeated across all participants and the result reported as mean and standard error across participants. Note that all 'distance-to-goal' measures refer to the shortest path to goal rather than the number of steps actually taken by the participant to reach the goal.

Analysis of hippocampal replays

For our analyses of hippocampal replays in rats, we used data recently recorded by Widloski and Foster (2022). This dataset consisted of a total of 37 sessions from 3 rats (n = 17, 12, 8 sessions for each rat) as they performed a dynamic maze task. This task was carried out in a square arena with 9 putative reward locations. In each session, six walls were placed in the arena, and a single reward location was randomly selected as the 'home' well. The task involved alternating between moving to this home well and a randomly selected 'away' well. Importantly, a delay of 5-15 s was imposed between the animal leaving the previous rewarded well before reward (chocolate milk) became available at the next rewarded well. On the away trials, the emergence of reward was also accompanied by a visual cue at the rewarded well, informing the animal that this was the reward location. In a given session, the animals generally performed around 80 trials (40 home trials and 40 away trials; Figure S6). For further task details, we refer to Widloski and Foster (2022).

For our analyses, we only included trials which lasted less than 40 seconds. We did this to discard time periods where the animals were not engaged with the task. Additionally, we discarded the first home trial of each session, where the home location was unknown, since we wanted to compare the hippocampal replays with model rollouts during the exploitation phase of the maze task. For all analyses, we discretized the environment into a 5x5 grid (the 3x3 grid of wells and an additional square of states around these) in order to facilitate more direct comparisons with our human and RNN task. Following Widloski and Foster (2022), we defined 'movement epochs' as times where the animal had a velocity greater than 2 cm/s and 'stationary epochs' as times there the animal had a velocity less than 2 cm/s.

Replay detection

To detect replays, we followed Widloski and Foster (2022) and fitted a Bayesian decoder to neural activity as a function of position during movement epochs in each session, assuming Poisson noise statistics and considering only neurons with an average firing rate of at least 0.1 Hz over the course of the session. This decoder was trained on a rolling window of neural activity spanning 75 ms and sampled at 5 ms intervals (Widloski and Foster, 2022). We then detected replays during stationary epochs by classifying each momentary hippocampal state as the maximum likelihood state under the Bayesian decoder, again using neural activity in 75 ms windows at 5 ms intervals. Forward replays were defined as sequences of states which included 2 consecutive transitions to an adjacent state (i.e. a temporally and spatially contiguous sequence of three or more states), and which originated at the true animal location. For all animals, we only analyzed replays where the animal was at the previous reward location *before* it initiated the new trial (c.f. Widloski and Foster, 2022). To increase noise robustness, we allowed for short 'lapses' in a replay, defined as periods with a duration less than or equal to 20 ms, where the decoded location moved to a distant location before returning to the previously decoded location. These lapses were ignored for downstream analyses.

Wall avoidance

To compute the wall avoidance of replays (Figure 4B), we calculated the fraction of state transitions that passed through a wall. This was done across all replays preceding a 'home' trial (i.e. when the animal knew the next goal). As a control, we computed the same quantity averaged over 7 control conditions, which corresponded to the remaining non-identical rotations and reflections of the walls from the corresponding session. We repeated this analysis for all sessions and report the results in Figure 4 as mean and standard error across sessions. To test for significance, we randomly permuted the 'true' and 'control' labels independently for each session and computed the fraction of permutations (out of 10,000), where the difference between 'control' and 'true' was larger than the experimentally observed value.

This analysis was also repeated in the RL agent, where the control value was computed with respect to 50,000 other wall configurations sampled from the maze generating algorithm (Algorithm 1).

Reward enrichment

To compute the reward enrichment in hippocampal replays (Figure 4C), we computed the fraction of all replays preceding a 'home' trial that passed through the reward location. As a control, we repeated this analysis for the remaining 7 locations that were neither the reward location nor the current agent location (for each replay). Control values are reported as the average across these 7 control locations across all replays. This analysis was repeated for all sessions. To test for significance, we randomly permuted the 'goal' and 'control' labels independently for each session and computed the fraction of permutations (out of 10,000) where the difference between 'goal' and 'control' was larger than the experimentally observed value.

This analysis was also repeated in the RL agent, where the control value was computed across the remaining 14 possible goal locations (that were not the current location or true goal).

Behavior by replay type

To investigate how the animal behavior depended on the type of replay (Figure 4D), we analyzed home trials and away trials separately. We constructed a list of all the 'first' replayed actions $\hat{a}_1$, defined as the cardinal direction corresponding to the first state transition in each replay. We then constructed a corresponding list of the first physical action following the replay, corresponding to the cardinal direction of the first physical state transition after the replay. Finally, we computed the overlap between these two vectors to arrive at the probability of 'following' a replay. This overlap was computed separately for 'successful' and 'unsuccessful' replays, where successful replays were defined as those that reached the goal without passing through a wall. For the unsuccessful replays, we considered the 7 remaining locations that were not the current animal location or current goal. We then computed the average overlap under the assumption that each of these locations were the goal, while discarding replays that were successful for the 'true' goal. This analysis was performed independently across all sessions and results reported as mean and standard error across sessions. To test for significance, we randomly permuted the 'successful' and 'unsuccessful' labels independently for each session and computed the fraction of permutations (out of 10,000) where the difference between successful and unsuccessful replays was larger than the experimentally observed value.

This analysis was also repeated in the RL agent, where we considered all exploitation trials together since they were not divided into 'home' or 'away' trials. In this case, the control was computed with respect to all 14 locations that were not the current location or current goal location.

Effect of consecutive replays

To compute how the probability of a replay being 'successful' depended on replay number (Figure 4E), we considered all trials where an animal performed at least 3 replays. We then computed a binary vector indicating whether each replay was successful or not. From this vector, we subtracted the expected success frequency from a linear model predicting success from (i) the time since arriving at the current well, and (ii) the time until departing the current well. We did this to account for any effect of time that was separate from the effect of replay number, since such an effect has previously been reported by Ólafsdóttir et al. (2017). However, this work also notes that many of what they denote 'disengaged' replays are non-local and would automatically be filtered out by our focus on local replays. When fitting this linear model, we capped all time differences at a maximum value of $|\Delta t| = 15$ s to avoid the analysis being dominated by outliers, and because Ólafsdóttir et al. (2017) only observe an effect for time differences in this range. Our results were not sensitive to altering or removing this threshold. We then conditioned on replay number and computed the probability of success (after regressing out time) as a function of replay number. Finally, we repeated this analysis for all 7 control locations for each replay and divided the true values by control values defined as the average across replays of the average across control locations. A separate correction factor was subtracted from these control locations, which was computed by fitting a linear model to predict the average probability of successfully reaching a control location as a function of the predictors described above. The normalization by control locations was done to account for changes in replay statistics that might affect the results, such as systematically increasing or decreasing replay durations with replay number. To compute the statistical significance of the increase in goal over-representation, we also performed this analysis after independently permuting the order of the replays in each trial to break any temporal structure. This permutation was performed *after* regressing out the effect of time. We repeated this analysis across 10,000 independent permutations and computed statistical significance as the number of permutations for which the increase in over-representation was greater than or equal to the experimental value.

For the corresponding analysis in the RL agents, we did not regress out time since there is no separability between time and replay number. Additionally, the RL agent cannot alter its policy in the absence of explicit network updates – which in our model are always tied to either a rollout or an action. As noted in the main text, an increase in the probability of 'success' with replay number in the RL agent could also arise from the fact that performing further replays is less likely after a successful replay than after an unsuccessful replay (Figure S8). We therefore performed the analysis of consecutive replays in the RL agent in a 'crossvalidated' manner at the level of the policy. In other words, every time the agent performed a rollout, we drew two samples from the rollout generation process. The first of these samples was used as normal by the agent to update $\boldsymbol{h}_k$ and drive future behavior. The second sample was never used by the agent, but was instead used to compute the 'success frequency' for our analyses. This was done to break the correlation between the choice of performing a replay and the assessment of how good the policy was, which allowed us to compute an unbiased estimate of the quality of the policy as a function of replay number. As mentioned in the main text, such an analysis is not possible in the biological data. However, since the biological task was not a reaction time task, we expect less of a causal effect of replay success on the number of replays. Additionally, as noted in the text, if some of the effect in the biological data is in fact driven by a decreased propensity for further replays after a successful replay, that is in itself supporting evidence for a theory of replays as a form of planning.

Supplementary note on experimental and architectural choices

In this short note, we discuss some of the many architectural and modeling choices that went into our work. As is the case for much work in modern computational neuroscience, the space of models was vast – and larger than we could feasibly explore fully in a single paper. In what follows, we hope to provide some additional motivation for the choices that were made in the main paper and to provide additional intuition for the importance and effect of various architectural choices and hyperparameters in our work. This note is also unlikely to be exhaustive, but we hope that it will be useful both for the reader hoping to gain a deeper understanding of our work, and for those looking to draw inspiration from it in their own computational models.

Network size

The size of the network used in our work is of some importance. We show in Figure S3 that our key results hold across a range of different network sizes. However, as the network becomes larger, its model-free 'base policy' also becomes better – to the point where rollouts become less and less useful as there is less room for improvement from the base policy. Indeed, in the limit of an infinitely large network trained on an infinitely large dataset, we expect a perfect base policy and no rollouts. On the contrary, if the network gets too small, it is unlikely to be able to learn how to use the rollouts for policy improvement, and we again expect rollouts to be less useful. In both limits, we also expect the notions of 'large' and 'small' to depend on the complexity of the task in question. For the task considered in this study, we found substantial use of rollouts across a range of network sizes between 40 to 140, and we found that the frequency of rollouts tended to decrease with network size (Figure S9). We did not test any networks larger than 140 units due to computational constraints associated with the training of large networks. The exact range of sizes for which our results hold will also depend on the type of network used, with LSTMs likely to be similar to the GRUs used in this work, and vanilla RNNs probably needing larger networks for comparable performance.

Planning horizon

In this work, we assumed a constant planning horizon of $L = 8$ steps and showed in Figure S3 that our key results are robust to changes to this hyperparameter between 4 and 12 (we did not test values outside this range). We chose a value of 8 for the main paper since it seemed like a reasonable planning depth in our fairly simple task with a relatively small action space, and it is comparable to the planning depth estimated in other simple games (van Opheusden et al., 2021). It is worth noting that some aspects of our results do change with planning depth. In particular, the change in policy for successful and unsuccessful rollouts is dependent on L, such that longer planning horizons lead to smaller average policy changes for successful rollouts and larger changes for unsuccessful rollouts. In the language of policy gradients, we expect that this is because the 'baseline' implicitly used by the network in its state update is related to the average success of a rollout. In other words, if almost all rollouts are successful (because L is sufficiently large), little is learned by observing that a rollout is successful, and the policy should not change much on the basis of this information. It is possible that there will be larger average policy changes in this setting if we instead condition on how late in the rollout the reward was found, which contains more information about the 'goodness' of the replayed trajectory. It would also be possible to make the planning horizon *variable* and let the agent itself choose its planning depth. This could be done in two different ways, namely by (i) making the agent decide up front how long a trajectory it wants to simulate, or (ii) letting the agent decide in closed-loop by iteratively returning a partial plan and deciding whether to continue planning or terminate and take a physical action. We opted for the simple fixed length solution since it has a smaller action space and fewer network iterations, making optimization easier. However, we expect that a variable planning length model is closer to real human behavior and believe that this will be an interesting avenue for future research.

Time cost of acting and planning

Related to the discussion of planning depth, we also assumed a constant temporal opportunity cost of planning for the agent. This was done despite the rollouts having variable length depending on whether and when the goal was reached during the rollout. We did this because the agent did not know *a priori* how long the rollout would be and had no direct control over its length. In the case of hippocampal replays being contained in sharp-wave ripples (SWRs), this is consistent with an assumption that a single trajectory occurs in a single SWR, and that the inter-SWR interval is independent of the length of the replayed trajectory. We did not train any agents with a variable temporal opportunity cost but do not expect any substantial differences from our current results.

More specifically, we defined a rollout in the model to last 120 ms. This is similar to the duration of hippocampal replays reported in the literature (Kurth-Nelson et al., 2016). In contrast, a single model action was defined to take 400 ms. These values were not directly fitted to the human data, as all model hyperparameters, including the episode length and relative cost of planning and acting, were chosen before any analyses of human behavior to avoid overfitting. Instead, the relative cost of rollouts compared to actions in the model, $\beta_{\text{roll}} := \Delta t_{\text{rollout}} / \Delta t_{\text{action}} = 0.3$, was chosen such that there was regular use of rollouts in the task. The episode length $T = 50$ *actions* was chosen to facilitate training of the model. We then designed our human behavioral experiment in a way that allowed participants to take approximately the same number of actions in a given episode as the model, which motivated an episode length of $T = 20$ *seconds*. This implicitly defined the 'duration' of a model action as $\Delta t_{\text{action}} = 20 \text{ seconds}/50 \text{ actions} = 400$ ms. The duration of a rollout was then defined as $\Delta t_{\text{rollout}} = \beta_{\text{roll}} \times 400 \text{ ms} = 120$ ms. Since we did not explicitly fit these parameters to the data, there are likely to be a range of parameter choices that lead to better fits to the human data from this particular experiment. Similarly, there is most definitely a range of hyperparameters that lead to worse data fits. Indeed our goal was not to chase the lowest possible discrepancy from human response times, but rather to demonstrate the general concept that models with the ability to perform rollouts do so in similar situations to humans. This is also the reason that we focus on correlations in the paper rather than e.g. MSEs, since the model 'thinking times' can be stretched, compressed, and shifted to different extents by altering the model hyperparameters.

Policy used for planning

In our work, we assumed that the policy used within the planning loop (i.e. the policy from which actions were sampled during a rollout) was the same as the policy used for sampling actions when actually interacting with the environment. We did this both for simplicity of exposition and computation, and because we think it is likely to be a reasonable approximation to how humans plan. However, there is in theory nothing in our model that prevents the rollout policy from differing from the action policy. In this case, rollouts can still be used to estimate gradients of the future reward with respect to the hidden state, provided that the policy from which rollout actions are sampled is known. This could be done e.g. through the use of importance sampling for off-policy learning. Additionally, in the case of sequential replays, it is plausible that previous replays directly affect future replays, e.g. in a process of exploration. In our current model, there was no option to systematically explore, and previous replays only affected future replays through their effect on the base policy. In theory, it would also be possible to more systematically explore the state space using sequential replays, and indeed we did experiment with 'rollouts' corresponding to node expansions of more advanced search algorithms, which can similarly be used to drive improved decision making. More generally, it would also be possible to optimize the rollout policy explicitly for planning by differentiating through the rollout process. This is in contrast to the present work, where the rollout policy was tied to the base policy, and rollouts were treated as part of the 'environment'. This meant that there was no propagation of gradients to allow for explicit adaptation of the policy to be better for planning.

Feedback from planning

When performing a rollout, the agent received an additional input on the subsequent timestep consisting of (i) a flattened array of the simulated actions, and (ii) a binary input indicating whether or not the rollout reached the (imagined) goal. Another reasonable choice of feedback input would be to return the sequence of *states* instead of the sequence of *actions*, or potentially to return both. Our reason for favoring the action sequence was primarily that the action space is lower dimensional (4) than the state space (16), which means that the input dimensionality is much lower than it would have been for the state sequence, assuming a one-hot encoding. This does raise the question of where this action sequence would emerge in biological circuits, given that hippocampal replays are canonically assumed to contain spatial information. However, we consider it reasonable that this state information could be converted to information about the actions that would take you there. Instead of returning a binary input of whether the goal was reached, the rollout process could also return the output of a learned value function. We did experiment with returning both the binary 'goal' feedback and the imagined value function, and we found that the agent predominantly used the goal information in this case. We therefore chose to remove the learned value from the feedback to simplify the model. However, we imagine that returning a learned value function would be useful in more complicated tasks with multiple or non-binary rewards.

Stochastic environments and multiple goals

For simplicity, we assumed that the environment was deterministic and that there was only a single goal. However, our model could also be extended to the setting of stochastic environments and multiple or non-binary rewards. In the case of stochastic environments, the agent would still need to simulate a *sample* from the policy. The internal world model was already trained to generate a distribution over new states, and in stochastic environments, we would want to sample from this distribution instead of using the maximum likelihood next state. Provided that the agent has learned a well-calibrated distribution over state transitions, the resulting rollout should still provide an unbiased estimate of the gradient of expected future reward with respect to the hidden state. In the case of multiple goals, it would still be possible to use the agent as-is and return a binary indicator of whether the agent reached any (or each) goal. However, as noted above, it would also be possible to return a learned value estimate instead of the binary goal information. In cases where these goals do not lead to random teleportation, it could also be useful to let the rollout continue beyond the goal. We chose not to do so in the present work, since the transition after reaching the goal was entirely unpredictable, so the simulated action sequence beyond this point would not be informative of expected reward.

Space in which to plan

We chose rollouts to occur in the space of states and observations. More specifically, the agent had to predict the upcoming state $\boldsymbol{s}_k$, and a new observation $\boldsymbol{x}_k$ was constructed automatically from $\boldsymbol{s}_k$ during the rollout. An alternative would have been to directly learn to predict $\boldsymbol{x}_k$, which we decided not to do since the majority of the input was constant within an episode. However, in more general task settings, where the environment is more variable, it might be simpler to predict the input directly. Additionally, in partially observable environments, there is a weaker correspondence between states and observations, and rollouts would require samples from the distribution over possible observations. This could either be done directly in observation space or indirectly via some inferred (or known) latent state space.

We consider it likely that humans do not plan explicitly in pixel space and instead use some form of latent planning representation. In the present work, this was also the case to some extent, since the agent input was already an abstract representation. However, in future work, it could be interesting to use a learned latent space instead. This could e.g. be done by training an autoencoder to reconstruct the state and reward information as in the VariBAD model (Zintgraf et al., 2019). Alternatively, planning could take place in a latent space explicitly optimized to yield good plans as in MuZero (Schrittwieser et al., 2020). We did not experiment with any of these possibilities but believe that the results would be comparable to our present work. A major reason for our choice to implement planning in the space of state transitions is that performing high-fidelity rollouts in state space only requires the agent to learn a state transition function. As has been demonstrated in previous work, a transition function could feasibly be learned in a self-supervised manner (Whittington et al., 2020), allowing agents to learn how to plan with little task-specific information. Additionally, rollouts in state space have close parallels to hippocampal replays as detailed in Figure 4.

Choice of task

The task used for human behavioral experiments and RL agents differs somewhat from the task used for the hippocampal replay data. Notable differences include (i) the presence of 'away' trials in the rodent data instead of the teleportation step in the human data, (ii) the different maze sizes and wall configurations, and (iii) the presence of a forced delay between rewards in the rodent data. A natural question is thus why we did not match the human and RL task to the rodent task, which we could not change since it relied on previously published data. There were a few major reasons for our decision to use different tasks for the humans and RL agents compared to the rodent experiments. One is that the rodent task was not a reaction time task, meaning that there was a forced delay between consecutive rewards. If we introduced a similar delay in the human task, there would be no incentive to act fast. Unfortunately, since we do not have access to intracortical recordings from the human subjects, the speed of acting is the major signal we analyse from our human participants, and it is therefore necessary with a reaction time task. Of course we could still have included away trials and used similar arenas without enforcing a delay between rewards. However, we cared mostly about the 'home' trials and therefore saw no reason to make participants spend half their time on 'away' trials. Additionally, the smaller Euclidean maze used in the rodent experiments would likely have been too simple for humans and reduced our signal to noise ratio, since there would be less time spent thinking. A simpler task and arena might similarly be simple enough that our RNNs could solve it in a fully 'model-free' manner without relying on rollouts to the same extent as in the present work. It is interesting to note that such suboptimality is a key factor of our results, but we believe that this is representative of human behavior as well, where thinking is mostly utilized in scenarios where we do not already know what to do.

Regularizing time or energy

In our RL agent, we did not incorporate any explicit energy costs for either actions or rollouts. Instead, the only unit of 'cost' was time elapsed. We did this since the only explicit incentive to be efficient in our human task was that fast decision making and good actions left more time for collecting reward. It could of course be argued that there is also some energy cost associated with taking actions in our online task, but (i) this energy cost is likely to be negligible, and (ii) if we wanted to model such energy costs in the RL agent, it would require us to introduce an additional hyperparameter to convert between 'energy' and 'time'. We considered it more interpretable and robust to only operate in the space of time, and we also believe that this is representative of many tasks encountered in our daily lives.