PHILOSOPHICAL TRANSACTIONS A

royalsocietypublishing.org/rsta

Research

Cite this article: Tsividis P *et al*. 2026 Human-level learning of complex novel tasks as theory-based modelling, exploration and planning. *Phil. Trans. R. Soc. A* **384**: 20240529. https://doi.org/10.1098/rsta.2024.0529

Received: 14 January 2025
Accepted: 26 July 2025

One contribution of 18 to a theme issue 'World models in natural and artificial intelligence'.

Subject Areas:
artificial intelligence

Keywords:
artificial intelligence, cognitive science, human learning, theory-based reinforcement learning, video game learning, cognitive modelling

Authors for correspondence:
Pedro Tsividis
e-mail: tsividis@mit.edu
Joshua B. Tenenbaum
e-mail: jbt@mit.edu

†These authors contributed equally to the study.

Electronic supplementary material is available online at https://doi.org/10.6084/m9.figshare.c.8451473.

THE ROYAL SOCIETY PUBLISHING

Human-level learning of complex novel tasks as theory-based modelling, exploration and planning

Pedro Tsividis[1], Joao Loula[1,†], Jake Burga[1,†], Juan Pablo Rodriguez[1,†], Sergio Arnaud[2], Nate Foss[1], Andres Campero[1], Ajay Subramanian[3], Thomas Pouncy[4], Samuel J. Gershman[4,†] and Joshua B. Tenenbaum[1,†]

[1]MIT, Cambridge, MA, USA
[2]Deep Dive AI, Mexico City, Mexico
[3]NYU, New York, NY, USA
[4]Department of Psychology, Harvard University, Cambridge, MA, USA

SJG, 0000-0002-6546-3298; JBT, 0000-0002-0138-163X

Humans are remarkable in their ability to quickly learn to perform complex tasks. Reinforcement learning (RL) has long been proposed as a model of human learning, and while leading machine RL models have surpassed human expertise at many classic board games and video games, they require vast experience to learn successfully—none of today's algorithms accounts for humans' ability to learn so many different tasks so quickly. We study human learning on 90 simple yet challenging video games, showing how people learn most games within a few minutes. To explain this behaviour, we propose a strong form of model-based RL, which we call theory-based RL, because it uses cognitively grounded intuitive theories—rich, abstract, causal representations of objects, agents and their interactions—to explore and model an environment and plan effectively to achieve goals. We instantiate the approach in an agent called EMPA (the exploring, modelling and planning agent). EMPA matches human learning efficiency, generalizing robustly to new game situations and levels, as humans

do, and exhibiting similar exploration and learning dynamics. Our work points the way for future efforts to build more detailed behavioural models as well as more human-like learning of complex tasks in artificial intelligence systems.

This article is part of the theme issue 'World models in natural and artificial intelligence'.

1. Introduction

Games have long served as microcosms through which to compare the flexibility and speed of human and machine learners. Simple video games are particularly revealing: while model-free reinforcement learning (RL) systems inspired by basic animal learning processes [1–4] can learn to play many classic video games with a relatively simple neural policy network [5–12], humans learn much more quickly and generalize far more broadly [13]. Human players can reach a competent level of play—scoring much better than a random policy and making significant progress that generalizes to many game board variations—in just a few minutes or a handful of episodes of play [13]. Model-free RL systems require tens or even hundreds of hours of play to reach the same level [7,12]. Even after reaching apparent mastery of a game, RL systems fail to generalize to even small variations of the game board or its dynamics [14].

Our goal here is to develop a computational account of how humans learn so efficiently and generalize so robustly. We seek to capture learning as we see it in humans, modelling the trajectories people show from the beginning to the end of learning a complex task such as a new video game. To do this, we start from the knowledge that humans bring to these tasks: flexible but powerful inductive biases—*a priori* preferences for some hypotheses over others—that are the product of evolution, development and culture, which can be deployed to learn quickly even in environments very different from their prior experience at the level of pixels or raw observations. The games we use in this study are designed to reflect key structural properties of real-world learning tasks: sparse and delayed reward, relational object dynamics and compositional goals that often require sequential subgoals to be achieved.

Our approach can be seen as a particularly strong form of model-based RL [15]—one that combines learnt symbolic dynamics models with structured planning and exploration objectives grounded in prior domain knowledge, rather than learning arbitrary transition functions from scratch. The broader idea of model-based RL has a long history in both human and machine learning [14,16–42], but as of yet there have been no proposals for how to capture human-level learning in complex tasks. In cognitive neuroscience, model-based RL has primarily been studied in simple laboratory tasks consisting of a small number of sequential decisions [43–45], where the algorithms and representations that are most successful do not directly transfer to the much more complex tasks posed by video games or the real world. In artificial intelligence (AI), model-based RL approaches are an active area of current neural network research and have the potential for greater sample efficiency relative to model-free systems [39–42]. But these approaches do not attempt to capture human inductive biases and are therefore still far from achieving human-level learning trajectories in most games.

We refer to our approach as theory-based RL, because it represents a solution to a standard RL task of taking actions to maximize cumulative reward, but one that explicitly attempts to incorporate ways in which humans, from early childhood, are deeply guided by intuitive theories in how they learn and act—how they explore the world, model its causal structure and use these models to plan actions that achieve their goals [21–23,46,47]. Exploration stems from theory-based curiosity, an intrinsic motivation to know the causal relations at work in the world and to take actions directed to reveal those causes [20,48–51]. Causal model learning can be formalized as Bayesian inference over probabilistic generative models [52], guided by priors based on intuitive theories—especially core knowledge representations dating to infancy that carve the world

into a natural ontology of objects, physics, agents and goals [17–19,53]. Planning exploits intuitive theories to constrain the search over an otherwise intractably large space of possible action sequences; knowing which future states are worth thinking further about helps even young children guide their search for effective actions and explanations [54,55]. All of these learning and reasoning abilities build on foundational perceptual capacities present from early infancy for detecting, tracking and individuating objects and events, across space and time, from vision and other sensory streams [16,53]. To illustrate, in one of our games ('Bait'), players must push boxes into holes to reach a key—inferring that holes are impassable, that boxes can fill holes and that the key enables progress. EMPA (the exploring, modelling, planning agent) uses its theory-based exploration and planning mechanisms to discover this through structured interaction and model building.

We embody this general theory-based RL approach in a specific architecture that learns to solve new game tasks, which we call EMPA (figure 1). We directly compare EMPA and alternative models with humans on a new large-scale behavioural dataset (available for download along with example videos of human and EMPA game play at https://github.com/tsividis/Theory-based-RL). The data consist of video game play from 300 human participants distributed across a set of 90 challenging games inspired by (and some directly drawn from) the General Video-Game AI (GVGAI) competition [56]. These games vary in ways that mirror real-world tasks: some require fast reactive play, while others require problem-solving with non-trivial exploration and long-range planning, and most feature very sparse rewards. Perceptually, these games are extremely simple: all objects are by default the same size and shape and can be represented effectively as 'super pixels' varying only in colour and location. This allows us to focus on the fundamental cognitive challenges of exploring, modelling and planning, rather than additionally explaining humans' ability to perceive in two-dimensional environments. People typically learn to play these games and generalize across levels in just a few minutes, or several hundred in-game steps—where each step corresponds to a discrete action taken by the agent or human player (figure 2). By comparison, as we show below, conventional deep RL algorithms typically take many thousands of steps to reach comparable performance on these games and often fail to solve even a single game level after hundreds of thousands of steps. EMPA aims to close this gap.

2. Learning, exploration and planning

EMPA interacts with the environment in a continuous cycle of exploring, modelling, planning and acting. It receives structured symbolic input extracted from the game state—including object classes, positions and discrete interactions—rather than raw pixels. These symbolic states form the input to all components of the agent. The *modelling* component learns symbolic program-like descriptions of game dynamics, including types of entities, causal laws governing the interaction between entities and win and loss conditions, as well as other sources of positive or negative rewards. Learnt models support the simulation of future states conditioned on a current (real or imagined) game state and the agent's actions. The *planning* component searches efficiently for plans that achieve goals, guided by theory-based heuristics that decompose win and loss conditions into reachable subgoals and goal gradients that reward steps towards those subgoals. The *exploring* component generates theory-guided exploratory goals for the planner, so that the agent most efficiently generates the data needed to learn a game's causal interactions and win or loss conditions. The electronic supplementary material, figure S1, illustrates how these components allow the agent to learn dynamics models rapidly, even in games that pose severe exploration challenges, and to immediately generalize plans to win new game levels, even with substantially different object layouts compared with those encountered during initial learning.

To make EMPA's architecture more concrete, consider how it solves the game Frogs (electronic supplementary material, figure S1). The agent begins with minimal knowledge of the environment but generates exploratory goals based on novelty and uncertainty—such as reaching an unfamiliar object across a river. It uses a symbolic model to track object identities, locations and

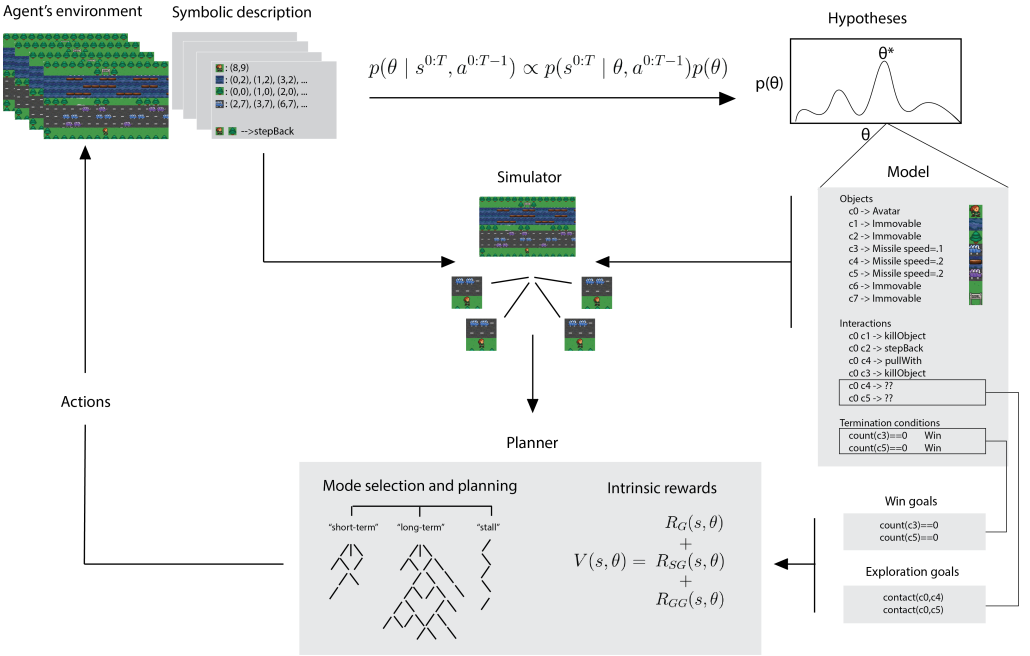

Figure 1. Schematic of the EMPA architecture. EMPA takes as input a symbolic description of the environment state, specified in terms of objects, their locations and the interactions that occur between them—all derived from the game's internal state rather than raw pixels (see Methods). Bayesian inference scores candidate environment models (θ) on their ability to explain observed state sequences. Theory-based curiosity generates exploratory goals. The planner decomposes exploratory and win-related goals into subgoals and goal gradients and uses this hierarchical decomposition to effectively find high-value actions for EMPA to take in the game environment.

simple contact interactions once they are observed (e.g. touching water causes loss; standing on logs prevents drowning). As it explores, EMPA incrementally learns the game's causal structure—discovering, for instance, that it can jump from log to log to reach the far bank, while staying safe from the dangerous water. Using its learnt model, EMPA performs symbolic planning to simulate candidate action sequences and select those likely to satisfy its current goals. In Frogs, this leads EMPA to coordinate a multi-step traversal across the river purely to test object interactions—a behaviour rarely seen in stochastic agents. The same planning and exploration mechanisms then allow EMPA to reach the win condition efficiently once the necessary steps are discovered.

EMPA integrates learning, planning and exploration in a unified framework, where each component is implemented using relatively simple, manually selected mechanisms. Planning is performed via best-first search over symbolic states, with search depth and memory limits chosen heuristically to balance efficiency and tractability (see Methods). Exploration is goal-directed, relying on structured novelty and expected information gain to prioritize interactions likely to improve the model. Learning occurs through incremental updates to a symbolic representation of dynamics, based on observed object interactions during gameplay. These choices were not tuned to individual games but reflected general principles expected to support rapid structure discovery in new environments.

Implementing EMPA requires many detailed technical choices (see Methods and the electronic supplementary material, *EMPA implementation*), but here we focus on the key ideas behind each of the components. The models learnt by EMPA are object-oriented, relational and compositional. They can be thought of as a projection of core intuitive theories (which allow infants to learn so rapidly about the real world) onto the virtual world of video games. Specifically, we represent models using a subset of the video game description language (VGDL; [57]), a lightweight language for games in which all the GVGAI games are written (see examples in the electronic

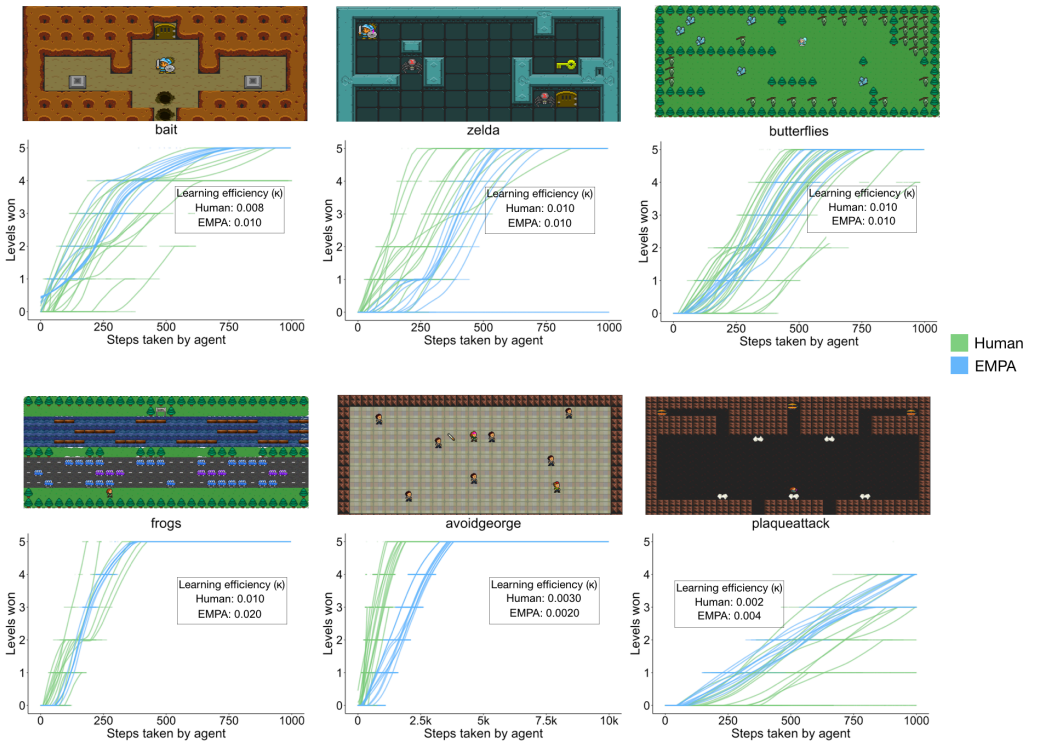

Figure 2. Humans (green) learn new games in a matter of minutes (corresponding to a few 100 steps, each taking 0.43 s on average), and EMPA (blue) closely matches human learning curves in these and many other games. Each dot shows the cumulative number of unique levels won at each point in time by an individual instance of each agent type (individual human or run of model); lines show smoothed scores from individual runs. Learning efficiency, $\kappa_{\text{levels}} = \frac{\text{levels completed}}{\text{levels in game}} \times \frac{\text{levels completed}}{\text{steps to completion}}$, compactly summarizes each agent's performance over the indicated range of experience (10 000 game steps for 'Avoidgeorge', 1000 for the remaining games).

supplementary material, figure S2). A VGDL description of a game specifies the appearance and basic properties of all objects, which manifest as causal constraints on their dynamic properties: whether they move by default and in what directions, how quickly they move, what their goals are and so on. Interaction rules specify the outcomes of contact events between pairs of objects as a function of their classes; these rules encompass natural concepts such as pushing, destroying, picking up and so forth. Finally, termination conditions specify when a game is won or lost. These transition dynamics, together with a description of the game's initial state, completely specify a given game world.

EMPA scores models according to a Bayesian criterion, with a hypothesis space corresponding to a restricted but still vast space of possible VGDL descriptions: for example, if we restrict to games with only 10 unique classes of entities, there are over 3.6×10^{37} possible VGDL descriptions or games that could be learnt. The posterior probability of a specific game description θ conditioned on observed game states s and actions a is

$$p(\theta \mid s^{0:T}, a^{0:T-1}) \propto p(s^{0:T} \mid \theta, a^{0:T-1})\, p(\theta). \tag{2.1}$$

The likelihood function, $p(s^{0:T} \mid \theta, a^{0:T-1})$, can be decomposed as

$$p(s^{0:T} \mid \theta, a^{0:T-1}) = p(s^0) \prod_{t=1}^{T} p(s_G^t \mid \theta_G, s_S^t, s_I^t) p(s_I^t \mid \theta_I, s^{t-1}, a^{t-1}) p(s_S^t \mid \theta_S, s^{t-1}, a^{t-1}), \tag{2.2}$$

which is a factorization of the state into goals, interaction events and non-interacting objects, respectively. Here, S_G refers to the part of the state representing termination status (e.g. win/loss conditions), S_I to object–object interaction events (typically involving contact or collisions) and S_S to the motion of independently moving objects not involved in interactions. These correspond to distinct components of the agent's model and allow modular learning and prediction. Further implementation details can be found in §7.

EMPA's exploration component implements a theory-based notion of curiosity that integrates its learning and planning functions, as the agent seeks to observe events that are most informative about unknown aspects of its hypothesis space. These exploratory goals can be formalized as a proxy reward function maximizing expected information gain [58,59]. Because of the nature of the models, the most informative events are always agent–object and object–object interactions for classes of objects that have not previously been observed in contact (see examples in the electronic supplementary material, figure S1). To determine these, the agent tracks a sparse count matrix of object pairings and actions and selects combinations it has not yet observed as candidate exploration goals. The agent thus explicitly makes plans whose goals are to produce these events. Crucially, this sense of curiosity is not myopic; EMPA often generates long-range plans whose sole purpose is to generate informative interactions that would be highly unlikely to occur by chance. For example, in Frogs (electronic supplementary material, figure S1), the agent crosses a dangerous river just to test interactions with previously unseen objects, despite no immediate reward. This enables EMPA to solve games that are highly challenging for traditional stochastic exploration methods.

EMPA's planner (see Methods for full description) uses the maximum posterior probability model as a simulator to imagine future states conditioned on candidate actions, searching a tree of future states to find high-value action sequences. Sparse external rewards are a fundamental challenge for RL agents; instead of relying solely on external rewards, EMPA assigns several kinds of imagined theory-generated rewards to imagined states. Treating exploration as a first-class goal, alongside winning, is one such theory-generated reward. EMPA also uses its domain theory to automatically decompose goals into easier-to-achieve subgoals, generated each time counts of objects thought to be relevant to winning are incremented or decremented, as well as goal gradients that help the planner move towards objects thought to be desirable and away from objects thought to be dangerous. To give a familiar example of how these concepts play out in real life, consider a task for an agent (human or robot) of watering all the plants in a room without slipping on the floor, such that the agent receives reward from the environment only upon completion of the entire task. The agent's goal may be formalized as making the number of unwatered plants equal to zero while avoiding contact with any puddle; a subgoal would then be to water a single unwatered plant, and a goal gradient would incentivize it to approach the nearest unwatered plant while staying as far as possible from puddles.

EMPA combines these terms in a single theory-generated reward function summing these three quantities, $V(s,\theta) = R_G(s,\theta) + R_{SG}(s,\theta) + R_{GG}(s,\theta)$ (for goals, subgoals and goal gradients, respectively), which can be applied to every environment, in all states s the agent encounters or imagines, for any model θ it currently considers most likely. EMPA's ability to auto-generate these rewards in terms of its abstract goal-based concepts, defined using the same description language as the agent's model learning and exploration, leads to a planner that effectively understands the agent's environments in a deep and generalizable way, and that is sufficiently powerful to find plans for most environments and tasks by simple best-first search through the space of imagined trajectories. The planner is also aided by iterative width (IW) pruning of insufficiently novel states from the search tree [60,61] and several different search modes specialized for different time scales of planning. A meta-controller manages decisions about whether to continue executing the current plan or search for a new one, as well as which planning mode to use. These decisions are governed by a small set of common-sense heuristics, based on the current game state, predicted state trajectory and model confidence. While these heuristics were not learnt, they approximate reasonable choices that could, in principle, be optimized or formalized further. The meta-controller calls

for long-term planning with deeper search trees when the environment is stable and predictable (e.g. when there are few stochastic objects in the game), short-term planning with shallower trees when the environment is rapidly changing or less predictable, and 'stall' planning when other modes have failed to return satisfactory plans.

3. Evaluating EMPA and alternatives versus human learners

We compared people and EMPA on a suite of 90 games (see Methods for human experimental procedures and a summary of the games; see the electronic supplementary material *GVGAI and variant game descriptions* for detailed descriptions of all games). We also tested a number of ablated versions of EMPA, designed to highlight the relative contributions of its modules, and two leading deep RL baselines, Double DQN (DDQN) [7] and Rainbow [12]. Deep RL methods provide a valuable comparison point for our work as they represent the class of most successful and actively developed RL agents on video-game tasks. We focus on DDQN, in particular, because it is a simple and widely known variant of DQN, the original deep RL system for playing classic (Atari) video games; we also evaluate Rainbow, a method designed to combine the best features of DDQN and other DQN variants.

Unlike deep RL agents, EMPA receives symbolic input: a structured representation of the game state in terms of objects, their properties and locations. This choice reflects our goal of modelling the cognitive mechanisms that support human-level learning—not of matching deep RL agents on perceptual inputs. Humans do not learn new tasks from pixels alone: they rapidly parse their environment into meaningful components like objects, agents and interactions. By assuming a similar structured input format, we isolate and evaluate the contributions of our core ingredients: modelling, exploration and planning. Symbolic input enables this focus but does not trivialize the problem—EMPA still must infer causal dynamics, explore efficiently and construct multi-step plans to win unfamiliar games.

We did not compare with deep RL agents operating over symbolic inputs, as this is not the context in which such models are designed to function. Their architectures are specifically tuned to process raw perceptual data, and adapting them to structured inputs would require significant re-engineering and extensive tuning—likely degrading performance rather than improving it. More importantly, our goal is not to match deep RL agents input-for-input, but to isolate and study the learning mechanisms that enable human-level efficiency: causal modelling, structured exploration and forward planning.

Our primary criterion for comparing humans and models is learning efficiency, which reflects both depth and speed of learning: How many levels of a game does an agent win, and how much in-game experience does it require in order to progress through these levels? Learning curves show intuitively that EMPA matches human learning efficiency across most games (figure 2; all 90 games are shown in the electronic supplementary material, figures S3 and S4). To more quantitatively compare human and model performance, we define a learning efficiency metric, $\kappa_{\text{levels}} = \frac{\text{levels completed}}{\text{levels in game}} \times \frac{\text{levels completed}}{\text{steps to completion}}$, a product of two terms measuring the degree to which a task is completed and the data efficiency with which it is accomplished. On balance, EMPA matches human learning efficiency according to this metric: it is sometimes better and sometimes worse, but almost always (on 79 out of 90 games) within an order of magnitude (0.1× to 10×) of human performance (figure 3). By contrast, DDQN almost always progresses much more slowly than humans: it is more than 100× less efficient on 67 out of 90 games, more than 1000× worse on 45 out of 90 games and over 10 000× worse on 22 out of 90 games. Rainbow, despite using a more sophisticated exploration policy [62] and being significantly more sample efficient than DDQN on Atari [12], performs no better on average here (figure 4). This is likely in part due to the fact that humans generally complete our tasks within 1000 actions, which is well within the margin during which both DDQN and Rainbow (and indeed all deep RL algorithms) are still exploring randomly. The fact that DDQN and Rainbow learn orders of magnitude more slowly

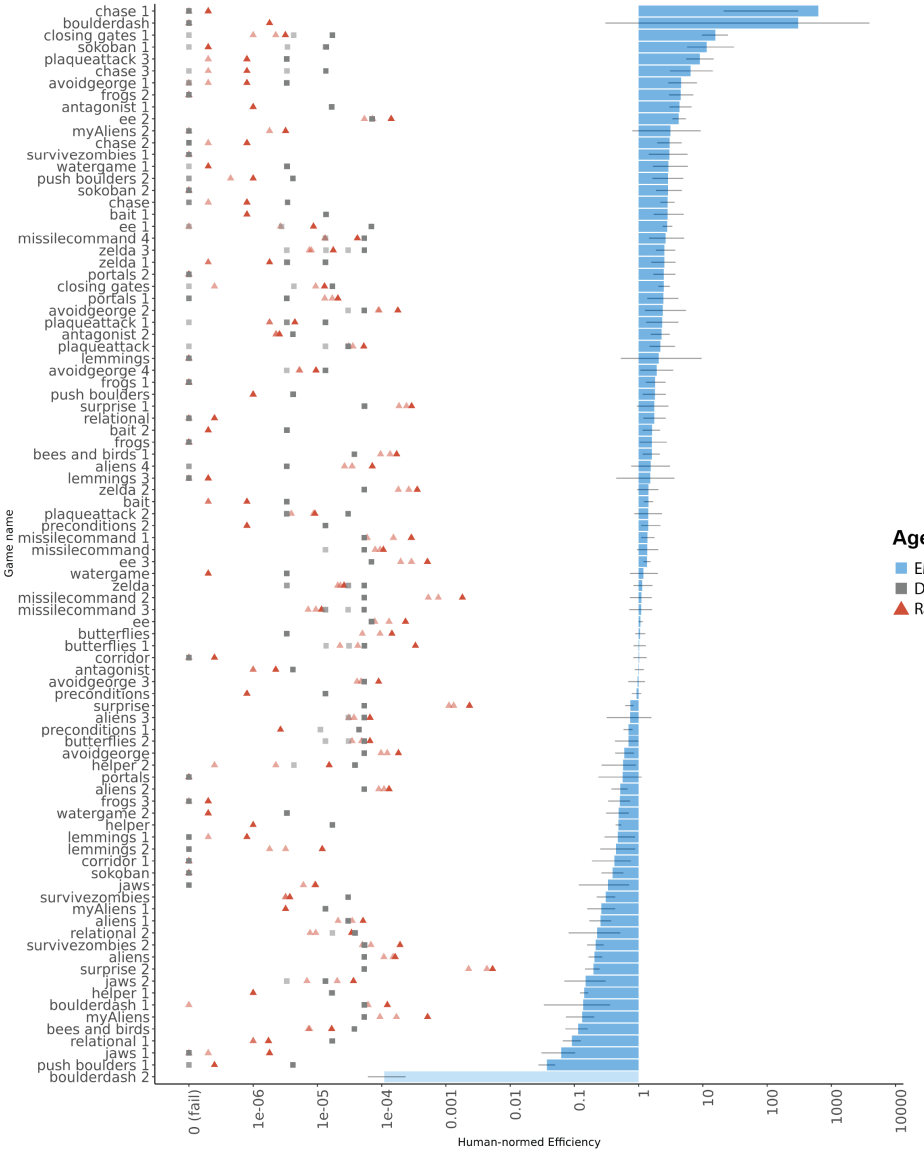

Figure 3. Across 90 games, EMPA achieves comparable learning efficiency to humans, outperforming them on roughly two-thirds of games and underperforming them on roughly a third, but is almost always within an order of magnitude (0.1–10×) of human performance. Shown in blue are mean scores and bootstrapped approximate 95% confidence intervals for 10 runs of EMPA on each game (adjusted in two games for sampling edge effects; see Methods for details). Shown in grey squares and red triangles are learning efficiencies for multiple runs of DDQN and Rainbow, respectively.

than humans should thus not be surprising: they are not designed with the objective of learning as quickly as possible. But this comparison does illustrate starkly the value of EMPA's more cognitively grounded cognitive capacities—a thousandfold gain in learning efficiency relative to these models and a close match to human learning speeds.

Although our game tasks are relatively simple, their goals and dynamics vary considerably in ways that reflect the complexity and variability of tasks humans solve in the real world. For instance, in Bait (figure 2, top left), the player must push boxes into holes in the ground in order to clear a path to a key that unlocks the exit. EMPA learns as fast as the median human, solving

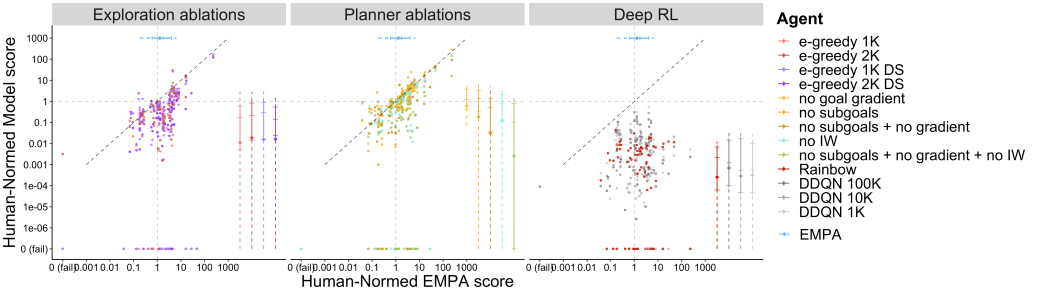

Figure 4. Ablations to EMPA's exploration (purple) or planning modules (yellow, turquoise and green) degrade the agent's ability to win many games, producing comparable failure rates to DDQN and suggesting that non-trivial exploration and planning abilities are a significant component of humans' ability to quickly perform well across many tasks. Each dot shows the performance of EMPA (*y*-axis) versus a comparison model (*x*-axis) on a given game. Line segments show 25th–75th percentiles (with medians marked by a central hash); dashed lines show 10th–90th percentiles; diamonds show geometric means.

all five levels in fewer than 1000 steps; DDQN wins the first level as fast as humans do, but takes 250 000 steps to solve the first two levels and fails to solve the rest within the allotted 1 million steps (see the electronic supplementary material, figure S4). In Zelda (top middle), the player must also reach an exit that only opens once the player has obtained a key, but here the player has to complete this task while avoiding rapidly moving, dangerous creatures. Humans and EMPA can solve this game within 500 steps, whereas DDQN takes 100 000 steps to win four levels and fails to reach the fifth in 1 million steps. A variant of the game (Zelda 1) that requires the player to pick up three keys before reaching the door is roughly equally challenging for humans and EMPA, who can win all levels within 2000 steps, but is far more challenging for DDQN—it takes 800 000 steps to reach the third level and never progresses beyond it, despite the similarity of the variant to the original. In Butterflies (top right), the player must catch randomly moving butterflies before they touch all the cocoons on the screen. EMPA performs as well as the best humans, winning all five levels within 1000 steps. DDQN solves the first four levels in as few as 20 000 steps, which is relatively fast compared with its performance on other games. However, it fails to solve the last level within 1 million steps, due to a subtle variation in the obstacle geometry that is fatal for the network's learnt policy but poses no problem to EMPA's or humans' model-based planning.

The modular nature of EMPA allows us to ablate different components and thereby gain insight into their relative contributions to the full system's success across our suite of games (see Methods for details). We compare full EMPA with variants in which planning heuristics are removed in different ways, as well as with versions in which exploration heuristics are replaced with stochastic exploration policies. We also compare against deep RL baselines: standard DDQN and Rainbow agents, trained either with the same limited sample budget as humans and EMPA (10k frames), or to convergence (1M frames; see Methods for details). Many games pose significant exploration challenges, requiring an agent to traverse a complex sequence of intermediate states (e.g. pick up a key to open a door, cross a road filled with dangerous moving cars or push several boxes into holes) in order to reach a state where the game's win conditions can be discovered. Such traversals are highly unlikely to occur by chance, and thus EMPA ablations that replace theory-based exploration strategies with ϵ-greedy exploration—essentially reducing exploration to stochastic exhaustive search—degrade performance on most games and cause catastrophic failure in others (figure 4). Likewise, ablations to any of EMPA's theory-based heuristics for planning (subgoal and goal gradient theory-generated rewards, or IW pruning) cause the system to perform worse in general, and in some games lead the planner to fail to discover solutions to even a single level within the allotted compute budget (24 h CPU time per game). Many games challenge the ablated planners for the same underlying reason that they challenge exploration: when multiple subgoals must be reached in a specific sequence before any reward is received, successful plans are hard to discover by a simple model-predictive search.

The ablated versions of EMPA still often outperform deep RL agents, despite lacking the agent's full exploration or planning strategies. This is because each ablation still retains critical cognitive ingredients that deep RL lacks. The exploration ablation still learns a structured, symbolic dynamics model and uses it to plan effectively—but explores using a stochastic policy. The planning ablation, by contrast, retains directed, theory-based exploration—but executes it using only shallow or greedy search. Both ablations can simulate future states and rapidly infer causal rules, constraining learning to a structured, low-dimensional hypothesis space. These capacities allow even partial versions of EMPA to generalize and act effectively in new tasks, far beyond what deep RL agents can achieve under similar conditions.

4. Fine-grained analysis of learning and exploring behaviour

EMPA matches humans not only in its learning efficiency but also in the fine-grained structure of its behaviour (figure 5A). While the specifics of effective agent trajectories vary greatly across games as a function of their object dynamics and layouts, humans and EMPA share consistent similarities: they generate short, direct paths to specific objects, in both 'explore' and 'exploit' phases of learning a new task, reflecting efficient plans to reach objects with potential (epistemic) or known (instrumental) value. By contrast, DDQN's stochastic exploration and slow learning generate behaviour that is much more diffuse and often restricted to a small subset of possible game board locations for long intervals (figure 5B). Unlike DDQN, human and EMPA trajectories are more efficient in later levels than previous ones (figure 5B) as they move from exploring the properties of new objects to exploiting their learnt models in order to win the game.

There are also significant differences between EMPA and humans, which in some cases lead to differences in behaviour. EMPA is able to move faster than most humans do and can thus more easily win in situations that require fast coordination. On the other hand, EMPA has weaker priors on higher level aspects of games, leading it to sometimes be more exploratory (and slower to win) than humans: for instance, EMPA does not know that games almost always have exactly one win condition, so after finding one way to win a game, it may continue to explore new ways of winning, whereas humans tend to win more efficiently by exploiting a single already-known win condition. Empirically, these differences tend to balance out and produce similar qualitative and quantitative behaviour, particularly when comparing EMPA and human performance with that of conventional deep RL agents.

In order to quantitatively compare EMPA's exploration, learning and planning with human behaviour, we classified objects across 78 games as 'positive' (directly results in a win), 'instrumental' (indirectly results in a win, e.g. a key that is needed to open the door), 'negative' (directly results in a loss) or 'neutral' (none of the above). We then coded how much time humans and model agents spent interacting with each class of objects. Across games, EMPA's profile of object interactions closely tracks the human profile (figure 5C), matching the distribution of interactions better than DDQN does, particularly in slow-paced games in which only the agent moves (see the electronic supplementary material, figure S5). DDQN spends the greatest proportion of its time interacting with neutral objects (e.g. walls), as these are most numerous across all games and therefore will be collided with frequently as a result of DDQN's random-exploration policy. By contrast, humans and EMPA display much more focused exploration that targets contact with object instances of unknown type as well as pairwise objects between other objects of unknown type (e.g. seeing what happens if it pushes a 'boulder' into a 'hole'); once these interactions are learnt, further contact occurs only if an object is positive or instrumental (implicated in win-related goals, as in a key needed to open a door), or neutral and on the path to a win, or if contact is unavoidable given the constraints of the game state. This behaviour reflects a kind of structured 'explore/exploit' behaviour, where early game play is characterized primarily by exploration and later play is characterized by exploitation of learnt rules and additional exploration when it is opportune.

In the real world, humans (and many other animals) learn quickly when a class of objects is dangerous and avoid them whenever possible. In our games, too, people typically interact with

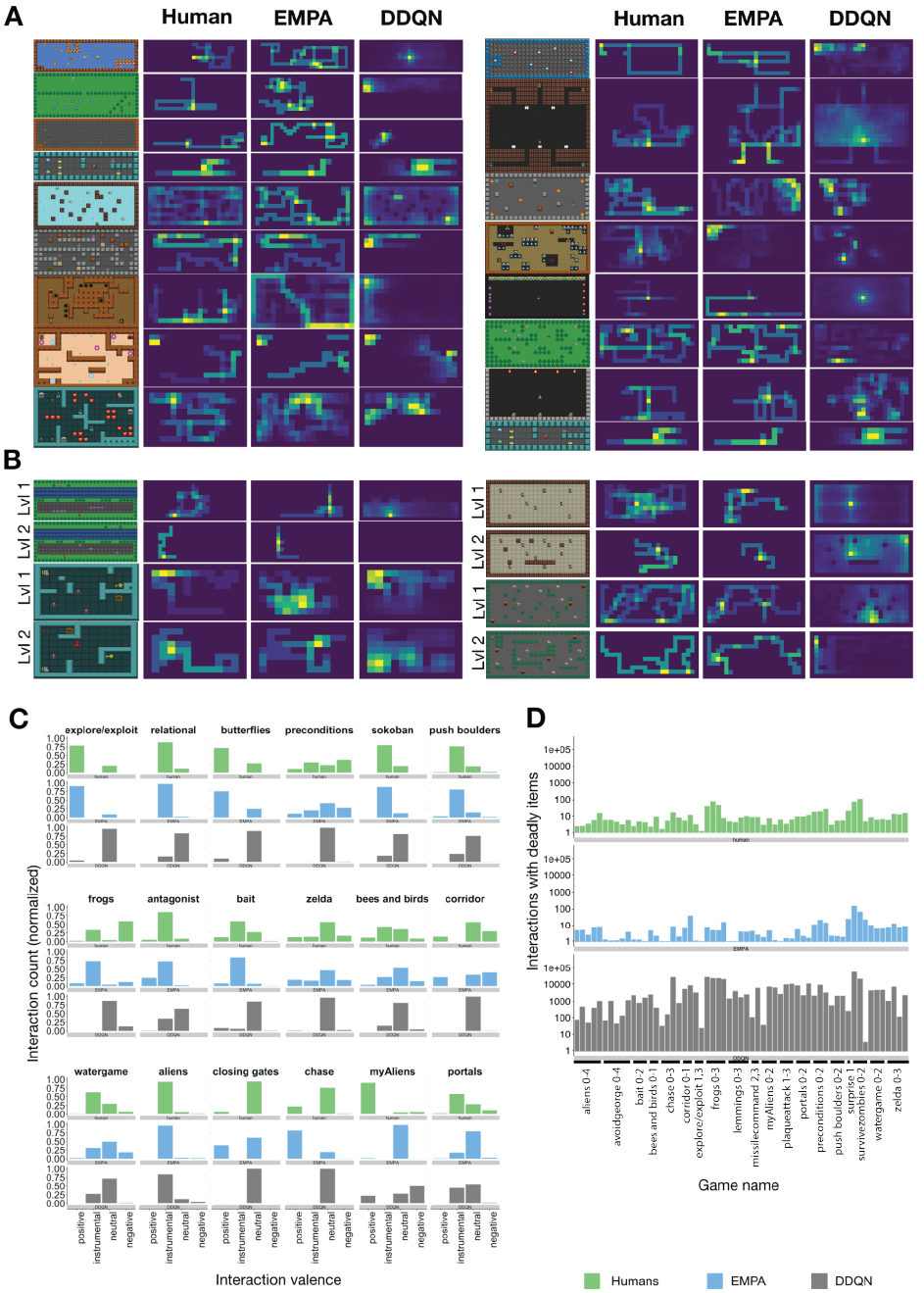

Figure 5. (A) Representative behaviour traces of individual human and EMPA players show qualitative similarities across a wide range of games. Heat maps show the time spent by each agent in each location of the game board for a given level, normalized by that agent's total experience on that level. Yellow indicates the most-visited locations; dark purple locations are not visited. (B) Humans and EMPA produce more diffuse trajectories during initial levels of games and more targeted behaviour in subsequent levels, once dynamics are known. (C) EMPA and humans interact with objects of different types in similar proportions; shown here are distributions of agent interactions with objects classified as 'positive', 'instrumental', 'neutral' and 'negative', for 18 representative games. (D) Humans and EMPA learn quickly from interactions with items that result in immediate loss and interact with such objects very few times.

objects that result in immediate loss only a handful of times across all episodes (figure 5D). EMPA behaves similarly and slightly more efficiently because of its rapid model-building and planning capacities, its strong assumption of causal determinism and its perfect memory and action execution: as soon as an object type is discovered to be dangerous, the planner avoids that object if possible. By contrast, DDQN typically interacts hundreds or thousands of times with deadly objects before it learns to avoid them (figure 5D). We do not currently analyse how these interactions change over time, but this remains a potential direction for deeper comparison.

5. Learning with an impoverished theory: extending EMPA to classic Atari games

Humans are able to succeed in a wide range of real-world environments despite having at best noisy, approximate internal models of physical dynamics [47,63,64]. By contrast, EMPA's hypothesis space (expressed in VGDL) can almost perfectly explain the dynamics of each of the environments (also expressed in VGDL) it encounters. This success raises the question of whether the principles of theory-based RL can generalize more broadly to environments outside of the agent's explicit hypothesis space.

To examine this question, we also test our approach on a small set of Atari games whose underlying dynamics are not written in VGDL or any similar language. The agent requires modifications to its learning component to support a more general state description and transition model, and to its exploring and planning components to support greater robustness to model–environment mismatch. As in the GVGAI experiments, it receives symbolic perceptual inputs; see Methods for a detailed description. We test the agent on five classic Atari games—Breakout, Carnival, Freeway, Pong and Space Invaders—and compare its performance with that of humans, Rainbow, EfficientZero ([65]; a leading model-based deep RL agent 500 times more sample-efficient than the original DQN), and a random policy. Across all five of the Atari games we studied, humans learn efficiently and effectively: they quickly outperform random play, and they reach reasonable scores within 6 min of game play (21K game frames, taking an action on average every 0.32 s). In four out of the five games (Breakout, Carnival, Pong and Space Invaders), EMPA's rapid learning efficiency is competitive with that of humans (figure 6; see also the electronic supplementary material, table S1). Performance improves at a similar rate and trajectory over the first 21K steps, reaching a mean score at 21K steps that is at least 75% of the mean human score (and indeed higher than the mean human score in two of the four games), and with a distribution of scores (figure 6) and learning efficiencies (electronic supplementary material, table S1) that significantly overlaps the human distribution in all four games. By contrast, the most efficient deep RL methods are not statistically distinguishable from random play, in either score distributions or learning efficiencies (figure 6 and the electronic supplementary material, table S1; see also figure S9 for two alternative and reinforcing analyses of comparative learning performance).

The last game—Freeway—poses unique challenges for all models. Despite its visual simplicity, it requires precise timing, long-horizon planning and effective exploration in a sparse-reward setting. Reward is granted only upon successfully crossing multiple traffic lanes and collisions—which are frequent—resetting progress without causing a loss or decrementing the score. Both deep RL agents and EMPA struggle in this setting, with the deep RL agents failing to score a single point across all runs. EMPA, while learning much faster than deep RL, still performs significantly below human levels due to a modelling failure: the core interaction in the game—being hit by a car—falls outside EMPA's hypothesis space. Instead of causing local, intuitive effects after being hit horizontally (e.g. death, horizontal displacement), a collision results in the agent being transported vertically downwards. This effect violates EMPA's assumptions about causal structure. As a result, the agent repeatedly mispredicts the consequences of collisions, leading to failed plans and frequent replanning. While humans can flexibly adapt to such dynamics, EMPA's strong inductive biases—typically an asset—become a liability in this environment.

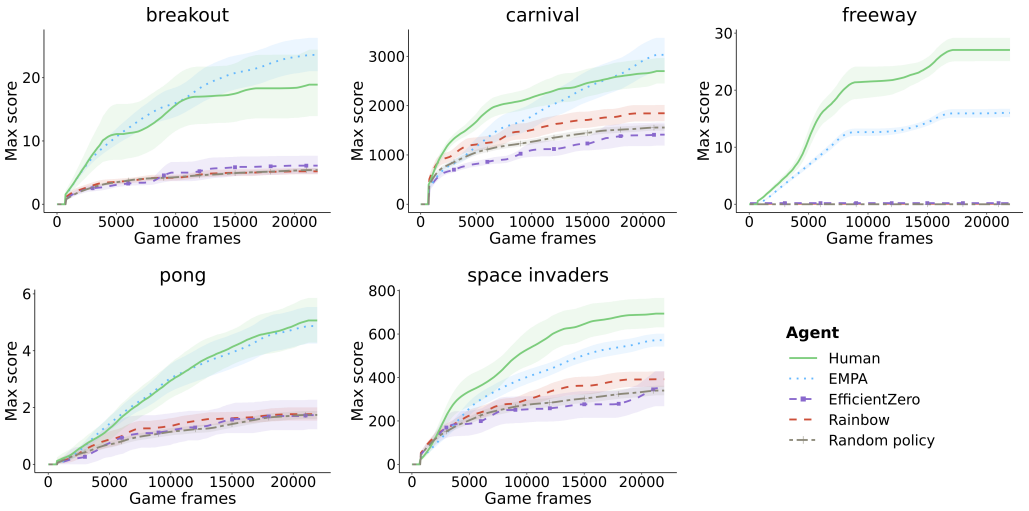

Figure 6. Comparison of humans, EMPA, deep RL and a random policy, learning to play five Atari games over 21K game frames (approx. 6 min of real-time play). EMPA's rapid learning efficiency is competitive with that of humans. Lines show the means of each model's maximum score at each point in time, averaged over multiple seeds. Shaded regions indicate approximate 95% confidence intervals.

EMPA succeeds at these tasks because after some exploration of each game, it learns a roughly accurate model of its environment. However, EMPA's model of these environments is not perfect (see the 'Model-environment mismatch in Atari' section in the electronic supplementary material for details on how it is unable to model the precise dynamics of each of the five games). As a result, the error of the agent's predictions increases from the time of any given prediction, reaching a mean error as high as 40 pixels per moving object after just 10 action steps on Pong (electronic supplementary material, figure S10A). These modelling mistakes, of varying magnitude and consequence, occur regardless of EMPA's experience—cumulative error throughout each game accumulates linearly with experience (electronic supplementary material, figure S10B). Still, EMPA is able to perform well by replanning when its prediction error crosses a threshold; usually, it only executes the first few steps of a plan before choosing to re-plan. Its predictions are good enough within this short time frame to allow it to reach human-level learning efficiency on most games and to far exceed state-of-the-art deep RL baselines.

6. Discussion

A long-standing goal of both cognitive science and AI has been to build models that capture not only what humans learn but how they learn. Humans' ability to learn efficiently and generalize flexibly has been a particularly important aspiration, and our work represents a first step in this direction by achieving human-level learning efficiency in a large set of simple but challenging and widely varying video-game tasks. Our work achieves this by incorporating inductive biases in the agent's model learning and exploration goals that directly implement proposals from computational cognitive science: model learning based on Bayesian priors, a hypothesis space based on objects, agents, physics, goals and causal determinism; exploration centred on pairwise interactions between objects and agents and a planner that takes advantage of the learnt model. The assumptions of causal determinism and uniformity that enable humans and EMPA to learn so quickly—assuming that interactions always produce the same effects, and objects of a given class have the same causal powers—are powerful and pervasive; even young children have these biases [66,67]. However, adults and children can also handle stochastic object interactions. EMPA could be extended to do so as well, using a Dirichlet prior to model transitions and resampling multiple

transitions in the planner before choosing the plan with the highest expected reward. EMPA's exploration module would need to be extended along the same lines to observe interactions multiple times before considering an interaction 'learnt'.

Our findings complement other recent work on theory-based RL [68–70]. Using a smaller set of VGDL games, one study [69] catalogued broad classes of human semantic and syntactic biases that can be incorporated probabilistically into a game-playing agent; in this work, we more explicitly elaborate on the 'core' nature of human inductive biases as well as the way these biases inform exploration and planning to explain rapid learning in complex environments. Another study [70], using functional magnetic resonance imaging (fMRI), reported neural signals associated with theory representation and updating that consistently tracked the mental events posited by EMPA. Taken together, these studies support our central claim that human learning of complex tasks is best explained by theory-based learning, modelling and planning. We note that our work is not intended to provide an account of humans' ability to learn continually, throughout life, from a single stream of experience—EMPA does not generalize from one game to the next, though extensions of the model could achieve this.

EMPA is also not directly fit to human behaviour, but its modular structure and symbolic representations make it a natural candidate for doing so. In future work, one could optimize components of EMPA—such as its exploration priorities, planning heuristics or model priors—to best fit individual human learning trajectories, enabling more traditional model comparison approaches. EMPA could also be extended to better capture human-specific constraints on planning and learning. This might involve modelling attentional bottlenecks, memory limitations or individual differences in inductive biases and exploration strategies. Such extensions would enable more fine-grained comparisons with human learning behaviour across tasks and trajectories.

In the real world, humans are able to succeed in a wide range of real-world environments despite having at best noisy, approximate internal models of physical dynamics [47,63,64]. By contrast, EMPA's hypothesis space (expressed in VGDL) can almost perfectly explain the dynamics of each of the environments (also expressed in VGDL) it encounters. This success raises the question of whether the principles of theory-based RL can generalize more broadly to environments outside of the agent's explicit hypothesis space. To examine this question, we tested our approach on a small set of Atari games whose underlying dynamics are not written in VGDL or any similar language. A modified version of EMPA was able to perform at human levels on five Atari games despite being unable to perfectly model the environments, suggesting the generality of the theory-based approach and the potential for extensions of the modelling approach to explain human behaviour in increasingly complex learning settings.

EMPA relies on the fact that the state space over which it operates is decomposed symbolically. Relaxing this assumption such that the challenge is to operate on a purely pixel-based environment requires modifying EMPA's inference procedure so that it can find the symbolic model that best explains the agent's non-symbolic sensory stream. The most direct way to accomplish this is hierarchically: inferring objects from pixels; inferring events from objects and inferring transition models from events and objects. The model presented in *learning with an impoverished theory: extending EMPA to classic Atari games* uses a version of this decomposed inference procedure, and we believe a more robust system could be built with a more significant engineering effort.

The fact that our agent uses a certain kind of model-based planner—a system for action selection that can directly and immediately exploit the agent's learnt model of objects in the environment and their interactions, and that takes exploration as an explicit goal—is both crucial to its success and grounded in cognitive science [68,71]. However, most specific choices made in implementing such a planner are not core aspects of our proposal; an investigation of the efficiency of the internal processing characteristics of EMPA and the comparison models and their relation to human thinking, whether measured in CPU time or power consumption, would be beyond our scope and methods. The planner incorporates a number of features that lead to increased computational efficiency and generality, relative to previous planners for video-game tasks, which allow it to solve all our tasks with a uniform compute budget. We view several of these features

as cognitively inspired, such as the combination of long-term, short-term and 'stall' modes, the meta-controller for switching between these modes and certain heuristics for planning around risky (unpredictable and dangerous) objects. These features represent potentially interesting hypotheses about how humans might think or plan efficiently that could be investigated in future work and tested against other successful planning methods [60,61,72,73].

Humans learn at least as much from others' experiences as from their own: in the domain of video games, watching an experienced player or reading instructions supports even more efficient learning about game dynamics or strategy than playing on one's own [13,74]. Additional recent work [75] uses the VGDL paradigm to address how humans accumulate knowledge across generations, showing that human learning trajectories when learning from each other with limited individual experience closely mirror the trajectories of individuals learning alone with unlimited attempts. Most intriguingly, humans can use their learnt models to achieve completely novel goals outside of those incentivized during training—an important aspect of children's play more generally that shows up specifically in how children approach video games [46]. For instance, a child might adopt the goal of losing (rather than winning) as quickly as possible, or of spending as much time as possible on a certain object without dying, or of teaching somebody else how the game works by touching each type of object exactly once. EMPA could be extended to play games in these ways as well, as its architecture makes it easy to specify arbitrary theory-generated goals for the planner.

In attempting to shed light on the inductive biases that make human learning so powerful, we were inspired by earlier model-based RL approaches based on object-oriented Markov decision processes (MDPs; [76]), relational MDPs [27] and probabilistic relational rules [28,29]. Our work more generally shares threads with recent methods that learn dynamics models giving objects, parts and physics first-class status [14,31], with methods that attempt more implicit unstructured model learning, including Bayesian RL [33–38,77], and methods that guide exploration by using language abstractions [78–80]. We go beyond such approaches by using explicit representations of physics, agents, events and goals more directly inspired by human psychology [16–20,48–51,54,55,68], and by exploiting these inductive biases in each of the three main components of our agent—exploration, modelling and planning.

Despite our demonstration of the value of building in strong inductive biases, however, we do not mean to suggest that computational approaches with less built-in structure could not be developed to achieve similar performance. On the contrary, we hope that our work will inspire other computational modellers and AI researchers to set this degree of rapid learning and generalization as their target and to explore how to incorporate—whether through deep model-based learning [39–42], meta-learning [81–84], simulated evolution [85], program synthesis [86–88] or hybrid neuro-symbolic or variational-Bayes object-centric architectures [89–94]—inductive biases like those we have built into our model. While our comparisons in VGDL focus on canonical model-free agents, newer sample-efficient deep RL models (e.g. EfficientZero, Dreamer v. 3 [95]) could also be evaluated on pixel-based renderings of these environments. We suspect that these models would show similar patterns of behaviour to what we observed with EfficientZero on Atari, but, nevertheless, view this as a valuable direction for future benchmarking. We expect that any system that eventually matches human-level learning in games or any space of complex novel tasks will exhibit—or at least greatly benefit from—a decomposition of the problem into learning and planning, and from inductive biases, theory-based exploration, planning mechanisms like those of EMPA and humans. Where this prior knowledge ultimately comes from in human beings, and to what extent it can be acquired by machines learning purely from experience or is better built in by reverse engineering what we learn from studying humans, remain outstanding questions for future work.

7. Methods

(a) Games and human behavioural procedures

Games were generated from a core set of 27 games, 17 drawn from the GVGAI competition [56] and 10 in a similar style that we designed. Each game comprised 4–6 levels, where different levels of a game use the same object types and dynamics but require different solutions due to their different layouts (board sizes, object counts and object configurations). For each of these 27 base games, we generated 1–4 variant games by altering different game features, such as the kinds of objects present, object dynamics and the game's win or loss conditions. Humans found these variants as interesting and challenging to play as they did the core set of games, and the games we designed did not significantly differ from the GVGAI-provided games in difficulty (electronic supplementary material, figure S6). All 90 games are described in the electronic supplementary material, *GVGAI and variant game descriptions*.

Three hundred human participants were recruited through Amazon Mechanical Turk and were paid $3.50 plus a bonus of up to $1.00 depending on their cumulative performance across all the games they played. Prior to playing the games, participants were given very minimal instructions: they were told that they should try to figure out how each game worked in order to play well, and that they could use the arrow keys and the space bar to play each game. For further details of our behavioural procedures, see the electronic supplementary material, *Experimental procedures*.

All games can be played online, using a Chrome browser, at http://pedrotsividis.com/vgdl-games/.

(b) Model learning: representation

EMPA builds models of each game environment using a subset of the VGDL [57]. VGDL specifies an environment in terms of three components corresponding to core aspects of human intuitive theories [46,96]:

Objects are a description of the appearance and basic properties of the different object classes in the game. These properties all manifest as causal constraints on the dynamic properties of the objects: whether they move by default, how and how often they move, whether they have simple goals (chasing or avoiding other objects) and if so, what those goals are. One object class is always the 'avatar', representing and controlled by the player. We call the set of these properties the *dynamic type* of each object. At every time step, VGDL uses the dynamic type to calculate proposed positions for each object. If the proposed positions are empty, the objects move into those positions. If they do not, VGDL handles the resulting collisions by looking at the rules specified in the Interactions.

Interactions are a list of rules that specify how pairs of objects interact to produce events. These events correspond to intuitively natural action concepts, such as pushing, destroying or picking up. All state changes beyond the movement patterns specified in the object-class descriptions are caused by these events. In turn, events are caused only by contact (collision) between objects.

For the sake of simplicity, EMPA only considers the following interaction types: `stepBack`, `bounceForward`, `reverseDirection`, `turnAround`, `wrapAround`, `pullWithIt`, `undoAll`, `collectResource`, `changeResource`, `changeScore`, `teleport`, `clone`, `destroy` and `transform`, but could easily be extended to learn additional interaction types and thus model a far greater space of games.

Termination conditions are a specification of the win and loss conditions for a game. We restrict our scope to games that are won or lost when counts of particular objects on the screen reach zero (e.g. `WIN IF count(BLUE)==0`).

A VGDL description can also be thought of as procedurally specifying a MDP consisting of the initial state of the game board, a transition function that specifies how the state evolves between

successive time steps (conditioned on the player's actions) and a reward function. The state at any time can be described by the object instances, their classes and locations; the avatar's internal state (the `agentState`), which tracks a set of resources the player has access to (e.g. health levels, or items carried) and any events occurring between pairs of objects that are participating in collisions at that time step. It is useful to decompose this state s into s_I, which refers to the pairs of objects that are participating in events as well as the nature of those events; s_S, which refers to the remaining objects and s_G, which refers to the `win/loss/continue` status of the environment. For the purposes of learning, we assume that the state space is fully observed.

Our concrete task is to learn a distribution over the set of possible VGDL models, Θ, that best explains an observed sequence of frames of gameplay. The agent does not store sequences of gameplay in memory across levels—instead, it simply maintains and updates its approximation of a distribution over possible models. For clarity, we decompose a particular model θ into three components corresponding to the objects (sometimes referred to as sprites), θ_S, interactions, θ_I, and termination conditions (or the agent's ultimate goals), θ_G.

The SpriteSet, θ_S, consists of dynamic type definitions for each unique object class in the game. For a given class c, its definition is a vector θ_{S_c} assigning values to each of the parameters needed to fully describe a VGDL sprite (see [57] for these complete descriptions). One of these parameters (`vgdlType`) always encodes the abstract type of the object, which places high-level constraints on its behaviour: whether it can move on its own or not, and if it can, whether it is an agent with chasing or avoidance goals, a random agent or a projectile. The remaining parameters specify details of the object's behaviour, such as its speed and orientation. The positions of sprites are updated at each step using simple programs that describe their behaviour. For example,

- `nextPos(Missile, speed = 2, orientation = Right, pos=(x,y)) = (x + 2, y)`
- `nextPos(Random, speed = 1, pos=(x,y)) = random.choice([(x,y), (x + 1,y), (x-1,y), (x,y + 1), (x,y - 1)])`.

These programs imply a distribution over next positions for any given parametrization; we use this distribution to calculate per-object likelihoods as explained in the electronic supplementary material, *EMPA implementation* (equation (7.2)).

The InteractionSet, θ_I, consists of rules, one or more for each pair of object classes, specifying the effects of those classes' interactions (see the electronic supplementary material, *EMPA implementation*, for further details).

The TerminationSet, θ_G, consists of termination rules, G, each of which causes the episode to end (transition to a `win` or `loss` state) if some condition is met; if no termination conditions are met, the episode continues. For simplicity, EMPA only attempts to learn termination conditions that can be expressed in terms of the count of objects in a given class equalling zero. This class of goals can capture the win and loss conditions of almost all games (for instance, the agent loses when the number of avatars goes to zero, or wins when the number of goal flags goes to zero), and reflects a standard abstract state feature in generalized planning [97,98]; it can easily be generalized if necessary.

(c) Model learning: Bayesian inference

The posterior probability of a model after a sequence of time steps, $1:T$, is $p(\theta \mid s^{0:T}, a^{0:T-1}) \propto p(s^{0:T} \mid \theta, a^{0:T-1})\, p(\theta)$. For simplicity, we use a uniform prior over theories. We present the mechanics of our inference procedure in the electronic supplementary material, *EMPA implementation*.

(d) Exploration

EMPA's exploration module works by setting epistemic goals for the planner, to observe the data most needed to resolve the learner's model uncertainty. Because the agent's knowledge is

encoded in a posterior distribution over a hypothesis space of simulatable models, an optimal exploration strategy could search over agent trajectories and select ones that, over some time horizon, maximize expected information gain (EIG; [58,59]) with respect to the agent's model posterior. However, the dynamic types of objects, θ_S, can be quickly learnt by observation and do not need to be explicit targets of exploration; only the interaction rules θ_I and termination conditions θ_G depend on the agent's actions.

Because of the object-oriented, relational nature of the model space, the uniform prior over interactions, the assumption that objects' interactions are determined completely by their classes, and the form of the likelihood function in equation (7.1) in the electronic supplementary material, seeing the events that occur after just one collision between objects of a given pair of classes c_j, c_k is highly informative about that pair's interaction rules, for all models θ in the hypothesis space. The module, therefore, sets exploratory goals of generating interactions between every pair of classes whose interactions have not yet been observed. In addition, because interactions between the avatar and other objects can vary as a function of the `agentState`, exploratory goals between the avatar and other objects are reset when the `agentState` changes (see the electronic supplementary material, figure S1, for an example).

Because of the likelihood function for termination conditions in equation (2.2) in the electronic supplementary material, the only states that are informative about termination rules for any hypothesis are ones where `count(`c_j`) == 0` for some class c_j. When the model learns that the count of some class can be reduced, it sets reducing its count to zero as an exploratory goal that allows it to learn about θ_G.

The initially large number of exploratory goals decreases as the model learns more about the game. At all points in the game, the planner mediates between exploratory goals and win-related goals as explained below.

(e) Planning: overview

The goal of the EMPA planner is to return high-value action sequences, together with the states predicted to obtain after each action. The details of the planner are cognitively inspired and were critical in building a working model of human behaviour in our tasks, but they do not constitute core claims on our part about how, exactly, people plan; we leave a detailed investigation of the heuristics with which people plan for future work. The EMPA planner takes as input a model, θ, and state, s, and searches action sequences using the procedure detailed in the electronic supplementary material, algorithm 1, *EMPA implementation*. Once a plan is found, it is executed until completion, or until the world state diverges sufficiently from the predicted state, in which case EMPA re-plans. We explain prediction-error monitoring and re-planning in the electronic supplementary material.

In order to plan efficiently in games with sparse rewards, EMPA evaluates plans in a hierarchical manner, with three distinct levels of representation: goals, subgoals and goal gradients.

Goals correspond to the fulfilment of known and hypothesized termination conditions, as well as to the fulfilment of contact goals specified by the exploration module. For example, if the agent's highest probability theory posits that the game is won by eliminating all objects of class c_j (e.g. the goal is to pick up all the diamonds), the planner will set as a goal `count(`c_j`) == 0`. Alternatively, before the agent has experienced any interactions with objects in c_j, the exploration module would set contact with any object of class c_j as a goal: `contact(avatar,` c_j`)`. The planner treats both exploration and win/loss goals equally, greedily fulfilling whichever it finds first. Exploration goals are active throughout the game, as long as they have not been fulfilled; often, the agent will be able to win all levels of a game without having explored every possible object therein.

Subgoals represent partial progress towards termination goals. Since all termination goals specify conditions on the size of a set of objects in a certain class (of the form, $|c_j| = N$), subgoals are defined to be any change in the number of instances of a relevant class that moves in the desired direction relative to the current game state. For example, if the agent currently has a goal to pick

up all diamonds, then any state that decreases the number of diamonds on the game board counts as fulfilling a subgoal.

Goal gradients represent preferences for states that are spatially closer to achieving a subgoal, computed based on the distances between pairs of objects in classes that are relevant to the agent's current goals. In our running example, a goal gradient expresses a preference for any action that moves the agent closer to the nearest diamond. But a goal gradient could also express a preference for bringing an object of one class closer to the nearest object of another class, if the agent has an exploratory goal to observe an interaction between those two classes. Goal gradients have 1% the weight of subgoals; they serve to guide the planner to more effectively explore physical paths and to prefer short paths over long ones. All goal gradients have the same weight regardless of the object type they are applied to.

Goals, subgoals and goal gradients are used both to define theory-generated rewards, supplementing the very sparse environmental reward structure, and to define the planner's completion criteria.

(f) Planning: theory-generated rewards

The agent's theory-generated reward function is a sum of the goal reward, subgoal reward and goal gradient reward: $V(s,\theta) = R_G(s,\theta) + R_{SG}(s,\theta) + R_{GG}(s,\theta)$.

Goal reward. All planner modes return a plan if a goal state is reached, and they stop searching states that stem from loss states, so $R_g(s,\theta)$ is effectively ∞ for `Win` states, $-\infty$ for `Loss` states and 0 otherwise.

Subgoal reward. By contrast, subgoal rewards only cause the short-term planner to return a plan, but they drive all planners' theory-generated rewards. Taking advantage of the fact that all termination goals specify conditions of the form $|c_j| = N$, states are penalized proportional to their distance from object-count goals. Specifically, the subgoal reward for a state s is

$$R_{SG}(s,\theta) = \rho \sum_{g \in \theta_G} \frac{N_{g_c} - |c(g)|}{|c(g)|^2} (-1)^{\mathbb{1}[g=Win]}, \tag{7.1}$$

where N_{g_c} is the class count specified by g and $|c(g)|$ is the actual class count in the state, of the class specified in g. ρ calibrates the relative value of subgoal reward to a goal gradient reward (explained below); we use $\rho = 100$.

Goal gradient reward. The numeric subgoal reward operates at the abstract level of object counts. Changes in this quantity are too sparse to guide search on their own, so the planner uses 'goal gradients' to score states at a lower level of abstraction by maximizing

$$R_{GG}(s,\theta) = \sum_{g \in \theta_G} \frac{d_{\min}(c'(g), c(g))}{|c(g)|^2} (-1)^{\mathbb{1}[g=Win]}, \tag{7.2}$$

where $c'(g)$ refers to the class (if one exists) that can destroy items of class $c(g)$, and $d_{\min}(c_j, c_k)$ refers to the distance between the most proximal instances of classes c_j and c_k. This biases the agent to seek proximity between objects that need to be destroyed and the objects that can destroy those objects, and vice versa for objects that need to be preserved.

The specifics of the goal gradients (that is, which particular objects to approach, avoid, ignore and so on) are generated by the planner within each game by analysing EMPA's highest probability model. For example, if the highest probability model has 'the agent wins if there are 0 diamonds on the screen' as a (paraphrased) termination condition, the planner finds all classes that, according to the model, can remove diamonds from the screen and gives high value to states in which such objects are near diamonds. More generally, if the highest probability model has $|c_j| = 0$ as a `win` condition for some class, c_j, the planner finds all classes c_k that the model believes can destroy c_j and generates goal gradients as specified above. This simply involves finding all

the c_k in rules (c_j, c_k, ξ) in which the affected class is c_j and the predicate, ξ, is one of `destroy`, `pickUp`, `transformTo`.

In principle, theory-generated rewards could be combined in various ways; our decision to sum them is one of convenience, rather than a claim about how humans plan. In terms of engineering, the critical factor is that goals contribute much more to the sum than do subgoals, which in turn contribute much more to the sum than do goal gradients. Similarly, equations (7.1) and (7.2) could be parametrized differently without changing the result, as long as they express the planner's general principles: 'prefer states in which the number of goal objects is closer to the desired number of goal objects', and 'prefer states in which objects that should collide are nearer to one another'.

In addition to these goal-gradient heuristics, the planner adopts several special ways of treating resources and projectiles that are not strictly necessary, but are helpful given the use of a restricted planning budget (see the electronic supplementary material, *EMPA implementation*).

Data accessibility. All behavioural and simulation data reported in this study are available at https://github.com/tsividis/Theory-based-RL.

Supplementary material is available online [99].

Declaration of AI use. AI-assisted technology was used to edit parts of the paper related to the reviewer response.

Authors' contributions. P.T.: conceptualization, data curation, formal analysis, funding acquisition, investigation, methodology, project administration, resources, software, supervision, validation, visualization, writing—original draft, writing—review and editing; J.L.: data curation, formal analysis, investigation, methodology, software, visualization; J.B.: data curation, software; J.P.R.: data curation, formal analysis, investigation, software, visualization; S.A.: formal analysis, software, visualization; N.F.: software; A.C.: investigation, software; A.S.: software; T.P.: investigation, writing—review and editing; S.G.: conceptualization, funding acquisition, methodology, project administration, resources, supervision, writing—review and editing; J.T.: conceptualization, funding acquisition, methodology, project administration, resources, supervision, writing—original draft, writing—review and editing.

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

Conflict of interest declaration. We declare we have no competing interests.

Funding. The work was supported by the Center for Brains, Minds and Machines (NSF STC award CCF-1231216), the Office of Naval Research (grant no. N00014-17-1-2984), the Honda Research Institute (Curious-Minded Machines) and gifts from Google and Microsoft.

Acknowledgements. The authors would like to acknowledge Aritro Biswas, Jacqueline Xu, Kevin Wu, Eric Wu, Zhenglong Zhou, Mark Goldstein, Alfredo Méndez, Bhavik Nagda, Michael Janner, Kris Brewer, Hector Geffner, Marty Tenenbaum and Alex Kell.

References

1. Sutton RS, Barto AG. 1981 Toward a modern theory of adaptive networks: expectation and prediction. *Psychol. Rev.* **88**, 135–170.
2. Schultz W, Dayan P, Montague PR. 1997 A neural substrate of prediction and reward. *Science* **275**, 1593–1599. (doi:10.1126/science.275.5306.1593)
3. Watkins CJCH, Dayan P. 1992 Q-learning. *Mach. Learn.* **8**, 279–292. (doi:10.1007/BF00992698)
4. Daw ND, O'Doherty JP, Dayan P, Seymour B, Dolan RJ. 2006 Cortical substrates for exploratory decisions in humans. *Nature* **441**, 876–879. (doi:10.1038/nature04766)
5. Guo X, Singh S, Lee H, Lewis RL, Wang X. 2014 Deep learning for real-time Atari game play using offline Monte-Carlo tree search planning. In *Advances in neural information processing systems*, pp. 3338–3346. La Jolla, CA: NeurIPS Foundation.
6. Mnih V *et al*. 2015 Human-level control through deep reinforcement learning. *Nature* **518**, 529–533. (doi:10.1038/nature14236)
7. Van Hasselt H, Guez A, Silver D. 2016 Deep reinforcement learning with double q-learning. *AAAI* **30**. (doi:10.1609/aaai.v30i1.10295)
8. Schaul T, Quan J, Antonoglou I, Silver D. 2015 Prioritized experience replay. *arXiv Preprint arXiv:1511.05952*

9. Stadie BC, Levine S, Abbeel P. 2015 Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv Preprint arXiv:1507.00814*
10. Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K. 2016 Asynchronous methods for deep reinforcement learning. In *Int. Conf. on Machine Learning*, pp. 1928–1937. Cambridge, MA: PMLR.
11. He FS, Liu Y, Schwing AG, Peng J. 2016 Learning to play in a day: faster deep reinforcement learning by optimality tightening. *arXiv Preprint arXiv:1611.01606*
12. Hessel M *et al*. 2017 Rainbow: combining improvements in deep reinforcement learning. *arXiv Preprint arXiv:1710.02298*
13. Tsividis P, Pouncy T, Xu J, Tenenbaum JB, Gershman SJ. 2017 Human learning in Atari. In *AAAI spring symposium series*. Palo Alto, CA: AAAI Press.
14. Kansky K *et al*. 2017 Schema networks: zero-shot transfer with a generative causal model of intuitive physics. In *Int. Conf. on Machine Learning*, pp. 1809–1818. Cambridge, MA: PMLR.
15. Sutton RS, Barto AG. 2018 *Reinforcement learning: an introduction*. Cambridge, MA: MIT Press.
16. Spelke ES. 1990 Principles of object perception. *Cogn. Sci.* **14**, 29–56. (doi:10.1207/s15516709cog1401_3)
17. Baillargeon R. 2004 Infants' physical world. *Curr. Dir. Psychol. Sci.* **13**, 89–94. (doi:10.1111/j.0963-7214.2004.00281.x)
18. Spelke ES, Kinzler KD. 2007 Core knowledge. *Dev. Sci*. **10**, 89–96. (doi:10.1111/j.1467-7687.2007.00569.x)
19. Csibra G. 2008 Goal attribution to inanimate agents by 6.5-month-old infants. *Cognition* **107**, 705–717. (doi:10.1016/j.cognition.2007.08.001)
20. Gopnik A, Glymour C, Sobel DM, Schulz LE, Kushnir T, Danks D. 2004 A theory of causal learning in children: causal maps and Bayes nets. *Psychol. Rev*. **111**, 3–32. (doi:10.1037/0033-295X.111.1.3)
21. Murphy GL, Medin DL. 1985 The role of theories in conceptual coherence. *Psychol. Rev*. **92**, 289–316.
22. Carey S. 1985 *Conceptual change in childhood*. Cambridge, MA: MIT Press.
23. Gopnik A, Meltzoff AN, Bryant P. 1997 *Words, thoughts, and theories*. vol. 1. Cambridge, MA: MIT Press.
24. Harris CM, Wolpert DM. 1998 Signal-dependent noise determines motor planning. *Nature* **394**, 780–784. (doi:10.1038/29528)
25. Kawato M. 1999 Internal models for motor control and trajectory planning. *Curr. Opin. Neurobiol*. **9**, 718–727. (doi:10.1016/s0959-4388(99)00028-8)
26. Fermin ASR, Yoshida T, Yoshimoto J, Ito M, Tanaka SC, Doya K. 2016 Model-based action planning involves cortico-cerebellar and basal ganglia networks. *Sci. Rep*. **6**, 31378. (doi:10.1038/srep31378)
27. Guestrin C, Koller D, Gearhart C, Kanodia N. 2003 Generalizing plans to new environments in relational mdps. In *Proc. 18th Int. Joint Conf. on Artificial Intelligence*, pp. 1003–1010. San Francisco, CA: IJCAI Organization.
28. Pasula H, Zettlemoyer LS, Kaelbling LP. 2004 Learning probabilistic relational planning rules. In *ICAPS*, pp. 73–82. Palo Alto, CA: AAAI Press.
29. Pasula HM, Zettlemoyer LS, Kaelbling LP. 2007 Learning symbolic models of stochastic domains. *Jair* **29**, 309–352. (doi:10.1613/jair.2113)
30. Xia V, Wang Z, Kaelbling LP. 2018 Learning sparse relational transition models. *arXiv Preprint arXiv:1810.11177*
31. Scholz J, Levihn M, Isbell C, Wingate D. 2014 A physics-based model prior for object-oriented mdps. In *Int. Conf. on Machine Learning*, pp. 1089–1097. Cambridge, MA: PMLR.
32. Keramati R, Whang J, Cho P, Brunskill E. 2018 Strategic object oriented reinforcement learning. *arXiv Preprint arXiv:1806.00175*
33. Oh J, Guo X, Lee H, Lewis RL, Singh S. 2015 Action-conditional video prediction using deep networks in Atari games. In *Advances in Neural Information Processing Systems*, pp. 2863–2871. La Jolla, CA: NeurIPS Foundation.
34. Fragkiadaki K, Agrawal P, Levine S, Malik J. 2015 Learning visual predictive models of physics for playing billiards. *arXiv Preprint arXiv:1511.07404*
35. Chiappa S, Racaniere S, Wierstra D, Mohamed S. 2017 Recurrent environment simulators. *arXiv Preprint arXiv:1704.02254*

36. Leibfried F, Kushman N, Hofmann K. 2016 A deep learning approach for joint video frame and reward prediction in atari games. *arXiv Preprint arXiv:1611.07078*
37. Ha D, Schmidhuber J. 2018 World models. *arXiv Preprint arXiv:1803.10122*
38. Racanière S *et al*. 2017 Imagination-augmented agents for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 5690–5701. La Jolla, CA: NeurIPS Foundation.
39. Kaiser L *et al*. 2019 Model-based reinforcement learning for atari. *arXiv Preprint arXiv:1903.00374*
40. Watters N, Matthey L, Bosnjak M, Burgess CP, Lerchner A. 2019 COBRA: data-efficient model-based RL through unsupervised object discovery and curiosity-driven exploration. *arXiv Preprint arXiv:1905.09275*
41. Schrittwieser J *et al*. 2020 Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature* **588**, 604–609. (doi:10.1038/s41586-020-03051-4)
42. Hafner D, Lillicrap TP, Norouzi M, Ba J. 2021 Mastering Atari with discrete world models. In *Int. Conf. on Learning Representations*. https://openreview.net/forum?id=0oabwyZbOu.
43. Daw ND, Gershman SJ, Seymour B, Dayan P, Dolan RJ. 2011 Model-based influences on humans' choices and striatal prediction errors. *Neuron* **69**, 1204–1215. (doi:10.1016/j.neuron.2011.02.027)
44. Otto AR, Gershman SJ, Markman AB, Daw ND. 2013 The curse of planning: dissecting multiple reinforcement-learning systems by taxing the central executive. *Psychol. Sci*. **24**, 751–761. (doi:10.1177/0956797612463080)
45. Kool W, Gershman SJ, Cushman FA. 2017 Cost-benefit arbitration between multiple reinforcement-learning systems. *Psychol. Sci*. **28**, 1321–1333. (doi:10.1177/0956797617708288)
46. Lake BM, Ullman TD, Tenenbaum JB, Gershman SJ. 2017 Building machines that learn and think like people. *Behav. Brain Sci*. **40**, e253. (doi:10.1017/S0140525X16001837)
47. Allen KR, Smith KA, Tenenbaum JB. 2020 Rapid trial-and-error learning with simulation supports flexible tool use and physical reasoning. *Proc. Natl Acad. Sci. USA* **117**, 29302–29310. (doi:10.1073/pnas.1912341117)
48. Xu F, Garcia V. 2008 Intuitive statistics by 8-month-old infants. *Proc. Natl Acad. Sci. USA* **105**, 5012–5015. (doi:10.1073/pnas.0704450105)
49. Schulz LE, Goodman ND, Tenenbaum JB, Jenkins AC. 2008 Going beyond the evidence: abstract laws and preschoolers' responses to anomalous data. *Cognition* **109**, 211–223. (doi:10.1016/j.cognition.2008.07.017)
50. Cook C, Goodman ND, Schulz LE. 2011 Where science starts: spontaneous experiments in preschoolers' exploratory play. *Cognition* **120**, 341–349. (doi:10.1016/j.cognition.2011.03.003)
51. Tsividis P, Gershman S, Tenenbaum J, Schulz L. 2014 Information selection in noisy environments with large action spaces. In *Proceedings of the 36th annual conference of the cognitive science society*, pp. 1622–1627. Quebec City, Canada.
52. Griffiths TL, Tenenbaum JB. 2009 Theory-based causal induction. *Psychol. Rev*. **116**, 661–716. (doi:10.1037/a0017201)
53. Spelke ES, *et al*. 2022 *What babies know: core knowledge and composition*. Oxford, UK: Oxford University Press.
54. Schulz L. 2012 Finding new facts; thinking new thoughts. In *Advances in child development and behavior* (eds F Xu, T Kushnir, J Benson). Oxford, UK: Academic Press.
55. Tsividis P, Tenenbaum JB, Schulz L. 2015 Hypothesis-space constraints in causal learning. In *CogSci...Annual Conference of the Cognitive Science Society. Cognitive Science Society (U.S.). Conference*. Austin, TX: Cognitive Science Society.
56. Perez-Liebana D, Samothrakis S, Togelius J, Schaul T, Lucas SM, Couetoux A, Lee J, Lim CU, Thompson T. 2016 The 2014 general video game playing competition. *IEEE Trans. Comput. Intell. AI Games* **8**, 229–243. (doi:10.1109/TCIAIG.2015.2402393)
57. Schaul T. 2013 In *Conference on Computational Intelligence and Games (CIG)*, pp. 1–8. Niagara Falls, ON, Canada. (doi:10.1109/CIG.2013.6633610)
58. Lindley DV, others. 1956 On a measure of the information provided by an experiment. *Ann. Math. Statist*. **27**, 986–1005. (doi:10.1214/aoms/1177728069)
59. Bernardo JM. 1979 Expected information as expected utility. *Ann. Statist*. **7**, 686–690. (doi:10.1214/aos/1176344689)

60. Geffner H, Lipovetzky N. 2012 *Width and serialization of classical planning problems*. Amsterdam, The Netherlands: IOS Press. (doi:10.3233/978-1-61499-098-7-540)
61. Lipovetzky N, Geffner H. 2017 Best-first width search: exploration and exploitation in classical planning. *AAAI* **31**, 3590–3596. (doi:10.1609/aaai.v31i1.11027)
62. Fortunato M *et al*. 2017 Noisy networks for exploration. *arXiv Preprint arXiv:1706.10295*
63. Battaglia PW, Hamrick JB, Tenenbaum JB. 2013 Simulation as an engine of physical scene understanding. *Proc. Natl Acad. Sci. USA* **110**, 18327–18332. (doi:10.1073/pnas.1306572110)
64. Gerstenberg T, Peterson MF, Goodman ND, Lagnado DA, Tenenbaum JB. 2017 Eye-tracking causality. *Psychol. Sci*. **28**, 1731–1744. (doi:10.1177/0956797617713053)
65. Ye W, Liu S, Kurutach T, Abbeel P, Gao Y. 2021 Mastering Atari games with limited data. *Adv. Neural Inf. Process. Syst*. **34**.
66. Schulz LE, Sommerville J. 2006 God does not play dice: causal determinism and preschoolers' causal inferences. *Child Dev*. **77**, 427–442. (doi:10.1111/j.1467-8624.2006.00880.x)
67. Gelman SA. 2003 *The essential child: origins of essentialism in everyday thought*. Oxford, UK: Oxford University Press.(Oxford Series in Cognitive Dev).
68. Pouncy T, Tsividis P, Gershman SJ. 2021 What is the model in model-based planning? *Cogn. Sci*. **45**, e12928. (doi:10.1111/cogs.12928)
69. Pouncy T, Gershman SJ. 2022 Inductive biases in theory-based reinforcement learning. *Cogn. Psychol*. **138**, 101509. (doi:10.1016/j.cogpsych.2022.101509)
70. Tomov MS, Tsividis PA, Pouncy T, Tenenbaum JB, Gershman SJ. 2023 The neural architecture of theory-based reinforcement learning. *Neuron* **111**, 1331–1344. (doi:10.1016/j.neuron.2023.01.023)
71. van Opheusden B, Kuperwajs I, Galbiati G, Bnaya Z, Li Y, Ma WJ. 2023 Expertise increases planning depth in human gameplay. *Nature* **618**, 1000–1005. (doi:10.1038/s41586-023-06124-2)
72. Helmert M. 2006 The fast downward planning system. *Jair* **26**, 191–246. (doi:10.1613/jair.1705)
73. Garriga Alonso A. 2017 Solving Montezuma's revenge with planning and reinforcement learning: Master's thesis. Universitat Pompeu Fabra, Barcelona.
74. Tessler MH, Goodman ND, Frank MC. 2017 Avoiding frostbite: It helps to learn from others. *Behav. Brain Sci*. **40**, e279. (doi:10.1017/S0140525X17000280)
75. Tessler MH, Madeano J, Tsividis PA, Harper B, Goodman ND, Tenenbaum JB. 2021 Learning to solve complex tasks by growing knowledge culturally across generations. See https://arxiv.org/abs/2107.13377.
76. Diuk C, Cohen A, Littman ML. 2008 An object-oriented representation for efficient reinforcement learning. In *25th Int. Conf. on Machine Learning*, Helsinki, Finland, pp. 240–247. New York, NY: ACM. (doi:10.1145/1390156.1390187)
77. Ghavamzadeh M, Mannor S, Pineau J, Tamar A. 2016 Bayesian reinforcement learning: a survey. *arXiv Preprint arXiv:1609.04436*
78. Tam A *et al*. 2022 Semantic exploration from language abstractions and pretrained representations. In *Advances in Neural Information Processing Systems 35*, vol. **35**, pp. 25377–25389, San Diego, CA.
79. Mu J *et al*. 2022 Improving intrinsic exploration with language abstractions. In *Advances in Neural Information Processing Systems 35*, vol. **35**, pp. 33947–33960, San Diego, CA.
80. Du Y, Watkins O, Wang Z, Colas C, Darrell T, Abbeel P, Gupta A, Andreas J. 2023 Guiding pretraining in reinforcement learning with large language models. *arXiv Preprint arXiv:2302.06692*
81. Wang JX, Kurth-Nelson Z, Kumaran D, Tirumala D, Soyer H, Leibo JZ, Hassabis D, Botvinick M. 2018 Prefrontal cortex as a meta-reinforcement learning system. *Nat. Neurosci*. **21**, 860–868. (doi:10.1038/s41593-018-0147-8)
82. Espeholt L *et al*. 2018 Impala: scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv Preprint arXiv:1802.01561*
83. Jaderberg M *et al*. 2018 Human-level performance in first-person multiplayer games with population-based deep reinforcement learning. *arXiv Preprint arXiv:1807.01281*
84. Dasgupta I *et al*. 2019 Causal reasoning from meta-reinforcement learning. *arXiv Preprint arXiv:1901.08162*
85. Salimans T, Ho J, Chen X, Sidor S, Sutskever I. 2017 Evolution strategies as a scalable alternative to reinforcement learning. *arXiv Preprint arXiv:1703.03864*

86. Ellis K, Morales L, Meyer MS, Solar-Lezama A, Tenenbaum JB. 2018 Dreamcoder: bootstrapping domain-specific languages for neurally-guided Bayesian program learning. In *Proc. 2nd Workshop on Neural Abstract Machines and Program Induction*. La Jolla, CA: NeurIPS Foundation.
87. Das R, Tenenbaum JB, Solar-Lezama A, Tavares Z. 2021 AutumnSynth: synthesis of reactive programs with structured latent state. In *Advances in programming languages and neurosymbolic systems workshop*. La Jolla, CA: NeurIPS Foundation.
88. Tang H, Key D, Ellis K. 2024 WorldCoder, a model-based LLM agent: building world models by writing code and interacting with the environment. See https://arxiv.org/abs/2402.12275.
89. Battaglia PW *et al*. 2018 Relational inductive biases, deep learning, and graph networks. *arXiv Preprint arXiv:1806.01261*
90. Chang MB, Ullman T, Torralba A, Tenenbaum JB. 2016 A compositional object-based approach to learning physical dynamics. *arXiv Preprint arXiv:1612.00341*
91. Watters N, Zoran D, Weber T, Battaglia P, Pascanu R, Tacchetti A. 2017 Visual interaction networks: learning a physics simulator from video. In *Advances in Neural Information Processing Systems*, pp. 4539–4547. La Jolla, CA: NeurIPS Foundation.
92. Zambaldi V *et al*. 2018 Relational deep reinforcement learning. *arXiv Preprint arXiv:1806.01830*
93. Garnelo M, Arulkumaran K, Shanahan M. 2016 Towards deep symbolic reinforcement learning. *arXiv Preprint arXiv:1609.05518*
94. Heins C *et al*. 2025 AXIOM: learning to play games in minutes with expanding object-centric models. See https://arxiv.org/abs/2505.24784.
95. Hafner D, Pasukonis J, Ba J, Lillicrap T. 2024 Mastering diverse domains through world models. See https://arxiv.org/abs/2301.04104.
96. Carey S. 2009 *The origin of concepts*. Oxford, UK: Oxford University Press.
97. Srivastava S, Zilberstein S, Immerman N, Geffner H. 2011 Qualitative numeric planning. *AAAI* **25**, 1010–1016. (doi:10.1609/aaai.v25i1.8013)
98. Bonet B, Francès G, Geffner H. 2019 Learning features and abstract actions for computing generalized plans. *AAAI* **33**, 2703–2710. (doi:10.1609/aaai.v33i01.33012703)
99. Tsividis P *et al*. 2026 Supplementary material from: Human-level Learning of Complex Tasks Through Theory-Based World Modeling, Exploration, and Planning. Figshare. (doi:10.6084/m9.figshare.c.8451473)