When Remembering and Planning are Worth it: On Navigation Strategies under Change

Omid Madani[1,4], J. Brian Burns[2,4], Reza Eghbali[3,4], and Thomas L. Dean[4]

[1] omidmadani@yahoo.com
San Carlos, CA
[2] Palo Alto, CA
[3] UC Berkeley, Berkeley, CA
[4] Brown University, RI, USA

Abstract. We explore how different types of memory can aid spatial navigation in changing uncertain environments. In our simple foraging task, every day, the agent has to find its way from its home, through barriers, to food. The world is non-stationary: from day to day, the location of some barriers or food may change. The agent's sensing is limited, and its location information is uncertain. Any map construction, and use such as planning, needs to be robust against such uncertainties. Any learning should be adequately fast. We look at a range of strategies, from simple to sophisticated, with various uses of memory. We find that the agent that builds and keeps updating a map, even though the map is partial and noisy, can be substantially more efficient than the simpler agents, as task difficulties such as distance to goal are raised, as long as the uncertainty, from localization and change, is not too large.

Keywords: Spatial Navigation, Foraging, Non-Stationarity, Autonomous Agents, Memory, Continual Learning and Planning, Sample Efficiency, Sample Bias, Planning under Uncertainty, Reinforcement Learning

"You can never know everything," Lan said quietly, "and part of what you know is always wrong. Perhaps even the most important part. A portion of wisdom lies in knowing that. A portion of courage lies in going on anyway."[5]
Winter's Heart, Book IX of the Wheel of Time, by Robert Jordan.

1 Introduction

The rich, productive, and ever changing world necessitates agents capable of continuous and flexible adaptations to achieve their objectives. In the world of engineered AI systems, reinforcement learning (RL) techniques [45, 20], specially the model-free variety using deep feed-forward neural networks, have had

[5] Excerpt in "The Bayesian Choice", by C. Robert [37].

substantial success in the past decade, as they are flexible, in that they do not assume much about the world, *e.g.* do not require modeling and encoding a complex world by the engineers of the agent, and powerful, as the (policy or value) function to be learned can be highly complex [41, 8, 31]. However, a critical limitation of the current model-free RL is the requirement of vast amounts of data. In dynamic real-world settings where the agent must adapt to evolving and changing tasks—ranging from minor adjustments to fundamental shifts—such systems often fall short: a change in the function (task) to be learned or performed can effectively reduce the available training sample size. Extensive and expensive simulations and substantial pretraining are common workarounds, striving to anticipate and train for *all possibilities*, as a way to enhance robustness, but the demands of the real productive world often curbs the success of such approaches. We seek efficient online learning and continual adaption that occurs ***in a lifetime, for a lifetime*** (*i.e.* keeping pace with life's changes and challenges).

Model-based techniques can potentially address some of the drawbacks of data inefficiency, their promise being that once acquired, the model(s) can be used repeatedly to find solutions for a diversity of related tasks. A challenge is what exactly an explicit model could mean, and whether such a model can be efficiently learned and updated. Consider spatial navigation, *e.g.* for foraging, which is a fundamental activity across the biological spectrum, with much research exploring how different minds meet its challenges [54, 42, 51, 3, 16, 22, 48, 32, 19]. Organisms need to be efficient (in energy/time) and flexible, and perform different but related navigation task (get to food, water, shelter, ...) in an at times dangerous and changing terrain (seasonal changes as well as abrupt changes, such as floods and droughts). The reward, from reaching the goal, is often distant, and the goal can change, thus the slow learning via reward propagation is often insufficient. From bacteria to bats and birds, living organisms utilize a range of sensing, memory, communication, and computational (e.g., inference) capabilities to efficiently reach their destinations [9, 48, 32].[6] While much of this machinery appears innate [22], a significant portion is likely dynamically and repeatedly learned and tuned and reconfigured throughout an organism's lifespan. In particular, the hippocampus is a structure that is established to be critically involved in memory formation and use, for instance to help the agent navigate via the creation of the so-called *cognitive maps* (such as place cells and grid cells) [33, 48], though the details, such as what is represented and how such is used, continues to be debated and investigated [53, 35, 12, 43].

A map data structure, once built, is versatile and highly useful for navigation: at least theoretically, to get from any point A to any point B (on the map), one can plan using the map and execute the plan, *i.e.* one can solve a range of (related) navigation tasks rather efficiently when one has access to a good map. The map can be modified and reused if reroutings are sought (*e.g.* in case of new

[6] Organisms of the same species, and the same organism but at different times, can use different (mix of) strategies [52, 18, 21]. The same person could use several strategies to get to a destination, *e.g.* from deciphering signs on a subway map to asking other people for directions (and remembering and executing those rough directions).

barriers blocking the routes that used to work). Consequently maps are the go to data structures for navigation tasks, *e.g.* in robotics and in particular SLAM domains [7]. But building a model (map) of a complex changing world, under limited time and sensing, carries its own many challenges. We explore these challenges in a simplified world and task: Consider a simple grid-world where every day an agent, with very limited sensing of its world, needs to find its way to food through barriers (road blocks). Moreover, the routes to food can change from day to day, some times substantially: several barriers or food may change location. A map could be part of a solution. If the world is fairly static, such a map could save much time over the lifetime of the agent. There are a number of challenges, including:

1. Any map learnt will be biased towards the experience of the agent, for instance, how much exploration it has performed (biased non-IID samples).
2. The world changes, and information extracted from the map can be outdated (uncertainty 1).
3. Agent's knowledge of its current location (we use simple path integration) contains errors due to motion (action) noise (uncertainty 2)
4. How (and whether) an agent would carve and granularize its sensed and perceived space into locations, to serve its needs, remains open.

We explore the first three challenges in this work. Issues of world complexity and substantial uncertainty has thwarted the practical use of model-based techniques for open-ended real-world tasks that contain a diversity of uncertainties. Probabilistic planning is highly intractable in general [27, 23], and earlier works on agents have also found that the focus on explicit representations and planning in traditional AI approaches may be misplaced, in part to due to the aforementioned challenges, and in part because simpler agent strategies can be sufficiently successful in a diversity of worlds and tasks [1, 2, 6].

In this paper, we investigate a number of navigation strategies, from simple to the more sophisticated, to see how they compare as we vary certain aspects of task difficulty: the environment size (distance to food), the proportion of barriers (path complexity), and two types of uncertainty: daily barrier/food location change and the uncertainty of localization (in agent location). In particular, each agent type uses a mix of (pure) strategies to get to food, and we compare the more sophisticated agents to a fairly simple mixed greedy agent (Table 1), that uses random action selection some of the time, together with the strategy of (greedily) lowering its distance to goal, at other times. A fly, for example, may execute a strategy akin to greedy via the use of the smell sense [28, 16].

In general, the simpler strategies require less (of memory and computing machinery)[7] and can remain useful in a wider range of environments: In environments where food is abundant and near, and obstacles are sparse, they would be

[7] Simpler strategies may rely on more sophisticated sensing: sensing itself can be compute intensive and involve pipelines of processing as well, for instance for estimating any of localization, orientation, and wind direction *e.g.* for smell [17, 28, 16].

adequate. However, in harsher more challenging and richer environments,[8] and when there exists (sufficiently stable) structure to the world, and enough time to make a decision (think, reason, ...), the more sophisticated strategies may outperform the simpler ones. We have two goals in this work:

- We ask: *Are the more sophisticated map-based strategies worth their costs (of memory, intricate control, and in general compute machinery)?* Under what conditions are they better than the simpler ones? By how much?
- (upon a positive answer wrt their worth) We provide insights on agent architecture and the types of memory and learning that could support efficient map construction and effective updates (map maintenance), and map use.

We find that well-designed memory-based strategies, that appropriately take uncertainties into account, in building, updating, and using memories that ultimately serve as maps (in this paper), outperform mixed-greedy and other simple strategies, and the advantage grows with environment size and difficulty (distance to goal and barrier portion), as long as the rate of change and location-uncertainty is not too large. Due to uncertainty and task complexity, pure strategies, such as always planning for a goal, may underperform drastically, and we find that robust behavior needs to use several pure strategies. Furthermore, a planning agent needs to take failure at planning time (*e.g.* no path to food) or execution failure (eg an unanticipated barrier) into account: Repeated replanning, as well as map updating, are necessities. Thus, there is indeed complexity to appropriate implementation of the more sophisticated strategies, but the gains in flexibility and reach can be worth it.

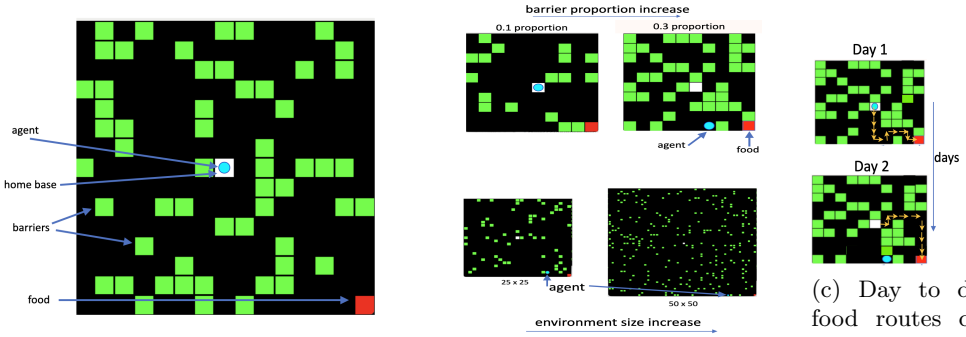
How does an agent (come to) know what to remember and how best to utilize its (episodic) memory? We assume certain capabilities, such as basic sensing, the importance of space and the details of path-integration, are (mostly) hard-wired [44, 11, 22]. In future work, we hope to reduce the number and the extent of the 'hard-wired assumptions' and in particular add additional learning in a lifetime (see Sect. 6).

This paper is organized as follows: We describe the simple (grid) environment and the basics of the agent and task(s) next. We then describe our flexible agent structure, which allows for incorporating multiple strategies, followed by describing the (pure) strategies, with different uses of sensing, memory, learning, and plannings, in Sect. 3. Sect. 4 presents our experiments, and Sect. 5 is on related work. We conclude with a summary and future directions in Sect. 6.

2 The Environment, Agent, and Task

Our environment is an N x N grid of N^2 cells, each cell identified by its (x, y). There is a single agent, with limited sensing (to be described). The agent is in exactly one cell of the grid at any time, and can execute a (move) action to change

[8] One view in philosophy of biology posits that the organism (agent) itself, with its capabilities and interactions, determines its own environment [29].



(a) The task.

(b) 2 ways of raising task difficulty.

(c) Day to day food routes can change.

Fig. 1: (a) The agent and its environment. It is important to emphasize that the agent *does not see* the whole grid, just the locations immediately adjacent to its current location (partial observability). (b) Two knobs on task complexity: barrier proportion and environment size (distance to goal). c) A 3rd knob on task difficulty: rate of (barrier) change, from day to day.

its location by one cell. Time is broken into day and time tick: day $1, 2, 3, \cdots$, and within a day, time ticks $t = 1, 2, \cdots$. We focus on a simple closed-world: a cell is in one of three states at any time: **EMPTY**, **BARRIER**, or **FOOD**. The agent has four actions: move to a single adjacent cell without BARRIER (**UP**, **DOWN**, **LEFT** or **RIGHT**). With ***motion-noise probability*** p, for a low $p \in [0, 1]$ (*e.g.* $p = 0.02$) the environment picks an alternative 'noisy' position, and this includes staying in the same location and moving two steps forward (Fig. 2(b)). An **illegal** action is one leading to a barrier and has no effect. The agent cannot go off the grid (assume barriers). Upon action execution, time tick is incremented.

Activity in the environment is broken into daily trips: in our experiments, each day, the agent starts at the home base, $(0, 0)$, and the task of the agent is to get to the food location, and save steps in doing so.[9] The state of a cell is not changed within a day, and there is always exactly one food cell.[10], and we ensure that the generated environments are such that a path to food exists. Once the food is reached, the next day begins and the agent location is reset to home $s = (0, 0)$. The food location does not change, *i.e.* the food is replenished, from day to day, unless there is an explicit change in the food's location (so the strategy of trying to go back to the same place food was found tends to be useful).

[9] In this work, we ignore costs of computation (in particular planning) costs (time or energy), and assume any computation by the agent is performed within the budgeted bounds.

[10] One cell with food can reflect a region that has food in practice.

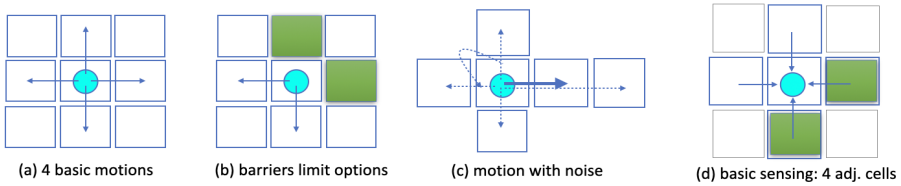


Fig. 2: Basic actions and sensing: (a) 4 possible actions: LEFT(west), RIGHT, UP, or DOWN. (b) In this example, with two barriers, the agent has two legal actions (left and down). (c) Motion noise, up to 6 possibilities: when intending to go east (right), with some (noise) probability, the agent may end up in another location: stay in the same cell, go up, or down, or left, or go two hops east. (d) Sensing is also from a single adjacent cell (4 such).

In most experiments, from day to day, we change several barrier locations, but keep the food in the same location (lower right corner).

Initially, on day 1, the agents doesn't know where the food is, nor the size of the grid, etc. (see the next section on limited sensing).[11] On the first day, and if (whenever) the food location changes, the task is more of a ***Search*** problem, and strategies geared toward exploration are more successful. The agent could remember certain aspects during each day to help it navigate and reach food in that day and in future days. Thus in future days, the task may become more of a path **Planning** problem, but only if the agent can remember the relevant aspects. Each agent type uses a different mix of (one or more) basic strategies, and different strategies involve different types of memory (*e.g.* short and long-term) and sensing (Sect. 3).

From one day to next, several barriers may disappear and some new ones may appear. In our experiments, we use the (barrier) **change-rate** to set this change: a change-rate of 0.1 means that about 10% of the previous days barriers are removed, and a similar number of new ones are added (overall barrier proportion kept the same).

2.1 Limited Sensing (Observing, Localizing, ..)

Animals use a variety of sense modalities for navigation, such as hearing (echo-location of bats), kinaesthesis, olfactory, and vision. In our work, we support a few basic sense capabilities and different strategies may use a subset of them. One available sense is looking one cell adjacent/around to get its state (FOOD, BARRIER, EMPTY) (a visual radius of 1). For the greedy strategy, we assume

[11] However, the strategies can be viewed as being designed for or having certain implicit/encoded assumptions in order to succeed in this type of task, such as 'keeping memories of what was observed at locations' is useful.

the agent has a sense akin to smell, telling the agent which of the 4 actions reduces distance to goal.[12] All strategies know which actions are legal.

The more sophisticated strategies require **localization**: access to an estimate $(\tilde{x}, \tilde{y})$ of the true current location (x, y). Our agent does simple **path integration** from its home $(0, 0)$, keeping two counters (sums), one for the horizontal, another for the vertical dimension. A RIGHT increments the horizontal counter, a LEFT decrements it (*e.g.* $(0, 0)$ becomes $(-1, 0)$), and so on.

Note that with a positive motion-noise p, the inaccuracy of the path-integration estimate $(\tilde{x}, \tilde{y})$ during the day is expected to grow with time tick t and in general the farther the agent is from its home base (starting point).

Dimensions of Difficulty. In some experiments we change the barrier portion or the grid size to change the difficulty of the task. For instance, a higher barrier portion means longer and more intricate paths to goals, and remembering where barriers are or the successful past paths can become more useful. On the other hand, increasing the barrier change-rate and motion noise can counteract the benefits of memory.

3 Strategies and Agents

An (autonomous) agent is a system that senses and acts so that it reaches or satisfactorily maintains certain internal states, which is reaching food for us (see Franklin et al [15, 14] for a good review of "agent" meaning). Even in our fairly simple task, we have found that stand-alone pure strategies rarely work (in a plausible diversity of environments). There is much evidence that organisms in nature use different strategies as well (eg allocentric, or map-based, vs sequential egocentric), and as a function of perception (not just time) [18, 21]. Thus we also explore *composite agents* in our experiments. As we present different (pure) strategies below, we touch on their strengths and weaknesses. Fig. 3 shows the basic loops of our agent, involving preparations, sensing and action execution, which we cover next.

3.1 Round-Robin Activation with Progressive Time Budgets

This round-robin setup makes it convenient to plug in various strategies and compare different combinations. At any timepoint, our composite agent executes the action selected by exactly one strategy, its **active strategy**. The agent sticks to its active strategy until its failure, or when its allotted time is up, in which case the agent moves to the next strategy on the list of strategies (user/experimenter specified), wrapping around in a round-robin manner, until food is reached (Fig. 3). For each strategy, the user can optionally specify a time budget. The agent begins each day with the given initial budgets. A strategy's budget, if specified,

[12] We do not model noise in the greedy direction (and one could raise such noise as the distance to food grows). On the other hand, smell can be more powerful and yield more information, such as the rough distance to goal (we do not model that either).

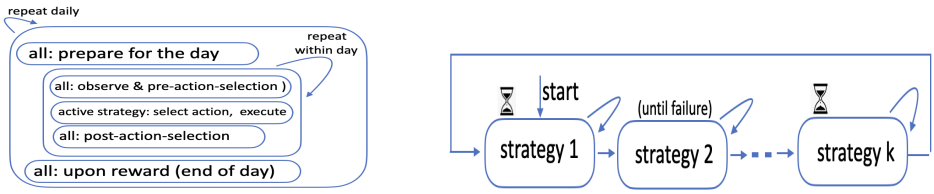


Fig. 3: (left) The control loop of a (multi-strategy) agent, generating its daily activity: the agent executes all strategies' pre and post functions, but uses exactly one, **active**, strategy that provides it with choice of action. (right) Our agent changes its active strategy at times, via a round-robin priority-ordered way: Each day the agent begins with using the first strategy. It moves to the next strategy (wrapping around), when current active strategy fails, or the strategy's time is up, until goal is reached (or all strategies fail). The allotted times (if any) are doubled, within a day, each time it starts the list over (Sect. 3.1).

is doubled, every time that strategy becomes active again in that day.[13] This is a simple low-memory way to keep the agent flexible, but using feedback and learning when, *e.g.* as a function of perception, and how much to use a given strategy is a future direction.

3.2 Interfacing with Several Strategies

In addition to the mandatory action selection function, a strategy can implement 4 other optional functions: pre-action-selection, post-action-selection, prepare-for-new-day (*e.g.* reset memories), and upon-reward. These allow the agent to provide information to the strategies, such as what cell is visited at current time and which action was selected. They also allow the strategies to store or do certain computations (*e.g.* memory consolidation). The action selection function is invoked only for the active strategy, but these four are invoked for *all* the strategies of the agent (once a day for the last two).

It is simple to incorporate bypassing, or support a subsumption architecture [5], in the above organization. For instance, if the food is seen, *e.g.* in the pre-action selection function, just go to it, ignoring the active strategy (which could be following a plan blindly), and execute the post-action selection (this bypassing lowers the number of steps somewhat).[14] We next describe the pure strategies we experiment with, roughly in order of increasing complexity. See also Fig. 4 which presents a summary of each strategy's requirements (localizing, memory, etc). The longer paper describes the strategies in further detail [26].

[13] If a strategy does not have a failure mode (*e.g.* Random) and has no time budget, once that strategy becomes active, it remains active for the rest of that day.

[14] In this specific scenario, the 'lower level' immediacy bypasses the 'higher level', while "subsuming" may imply the higher-level overriding the lower level.

3.3 The Strategies

Random. This baseline strategy returns a legal move picked uniformly at random (up to four possibilities). It requires no memory, but is highly wasteful. A small change, which we call ***Biased***, is a substantial improvement (Table 1). Biased does not take a 'step back' when possible, picking uniformly at random from the remaining legal actions (*e.g.* if LEFT was executed at $t-1$, RIGHT is not selected at t, unless it's the only legal action). This variant requires a bit of memory.

Greedy. The Greedy strategy has access to the action(s) that lead to lowering the distance to the goal (up to 2 such), and picks one such at random. Greedy can fail, *i.e.* barrier(s) can block those directions. Thus it cannot be used alone. Like Random, Greedy does not need any of the pre and post functions. A variant, **memory-greedy**, does not require a smell direction but requires the memory of the food location from the day before. If food is fairly static, it performs similar to greedy except when Search is required (*e.g.* 1st day).

Least-Visited. This is the first strategy that makes extensive use of what could be viewed as a type of episodic memory (medium-term, for a single day), which also requires localization. In its prepare-for-the-day function, it allocates an empty mapping, of location to visited count, and in its pre-action-selection function, increments the visit-count of its current location (location estimate, $(\tilde{x}, \tilde{y})$, via path-integration of Sect. 2.1). Whenever it is the active strategy, it picks an action that takes it to the cell with lowest visit-count, ties broken at random. At the expense of the memory requirement, this is a more efficient explorer, when Search is needed, compared to Random.[15]

Path. This strategy remembers yesterday's path, in the form of a mapping, from visited location s, $(\tilde{x}, \tilde{y})$, to last action taken (selected by any strategy) at s, via the post-action-selection function. When there is no change (or no new barriers and food hasn't changed), and no motion noise, the remembered path yields a successful path for the next day. Otherwise, there can be a few ***execution-time*** failure cases (in which case the agent should change strategy): (1) The path (mapping) returns an illegal action (*e.g.* due to new barrier) (2) Goal is reached, but no food (*e.g.* due to localization error) (3) Current location is not in the map (*e.g.* from use of other strategies).

Path is akin to model-free RL solutions in that the path can be viewed as a policy, mapping locations (state features) to actions (also, akin to sequential egocentric [18]). This strategy works for only one goal, and if there are multiple goals or destinations, the agent may need to learn different paths (different mappings, or functions), which can become both sample and space inefficient. We also experimented with DQN [31], which is model-free NN approach that

[15] Many organisms appear to have developed so-called Levy walks and jump strategies to more efficiently search a large expanse in finding clusters of food [54]. LeastVisited in conjunction with planning (ProbMap) could be used for such search as well.

strategy ↓	localization	smell	within day memory	multi-day memory	planning
Random	–	–	–	–	–
Greedy	–	✓	–	–	–
mem. Greedy	✓	–	–	✓	–
LeastVisited	✓	–	✓	–	–
Path	✓	–	✓	✓	–
ProbMap	✓	–	✓	✓	✓
DQN	depends	–	–	✓	–

Every day is a brand new day! (forgets experience)

Records/uses some memories, such as yesterday's food location.

Fig. 4: A summary of what different strategies use or require (mainly of the agent, but also of the environment). Greedy requires the smell ('gradient') direction. Localization, ie availability of the $(\tilde{x}, \tilde{y})$ estimate of the current location for the agent, need not be perfect (Sect. 2.1). DQN's long-term memory is in its neural-network weights, and its input vector includes current $(\tilde{x}, \tilde{y})$ in our experiments.

has been very successful in fully-observable (ATARI) game playing. We provided basic features (the surround) and location to the network (Sect. 4.2).

ProbMap. This (allocentric) strategy records and keeps updating memories of the barriers and food, and uses such memories to make (in effect) a map, that includes a start and goal, and plans a path to the goal. The output of the planning is same as for Path (a mapping from location to action). This strategy is the most elaborate and expensive in terms of the compute and control infrastructure it requires, but is the most powerful and flexible of the strategies, as the start and goal locations need not be fixed.

Two types of memories are maintained: (1) ***episodic*** (fast memory) (2) **statistical** (slow learning). Memories have different uses [30], but for our task, they are used to figure out where food and barriers could be, in particular, for planning. An episodic memory is an explicit record (tuple) of time (day and tick), location (via path-integration), and the object observed (EMPTY,...). For instance, the tuple $< (3, 5), (0, -1)$, BARRIER$>$ would mean a barrier was observed on day 3, tick 5, location $(\tilde{x}, \tilde{y}) = (0, -1)$. Each location is used as a key pointing to a list of such memory records (a hashmap). At every time point, the strategy updates these lists for each of the adjacent four locations (in the pre-action-selection). Such a memory could be used directly for planning. For instance, an agent could use only the latest memory (drop or ignore older ones). However, by learning the prediction distributions and performance, *i.e.* slow learning, one can do better.

Individual episodic memories have a sparsity problem: they don't occur enough to provide probabilities. The strategy converts them into ***memory types*** by dropping the location. Thus the question becomes 'how does a barrier seen yesterday predict (same location) for today?' (and not 'how does barrier at specific location (1,2) predict..?'). At every time point, ProbMap converts its existing episodic memories into memory types, and updates their 3-element distributions

and logloss performance, using EMA and other (window-based) sparse moving average techniques [24].

When planning, ProbMap collects locations that have sufficiently high food probability, and picks one at random. For barriers, for each location, memory-types that have the best logloss provide barrier probabilities, and it samples barrier locations using them. It does the planning, using A^* search.

Two types of failures are possible. If there is an execution time failure (*e.g.* new barrier) ProbMap replans. In case of plan-time failure, *e.g.* no food (goal) on the first day, the agent needs to change strategy.

Oracle. This strategy is meant to provide a reference point, *i.e.* the lowest number of steps on a given day and environment. The strategy always has the complete up-to-date map (of barriers and food), as well as the true location of the agent, and plans accordingly (similar to ProbMap, uses A^* for planning)

A Discussion of Types of Memory. There are many ways to classify memories, such as episodic, semantic, associative, working, short-term *vs.* long-term, internal *vs.* external (*e.g.* using notebooks), biographical, and so on. We noted that Biased uses a little memory, and control strategies such as round-robin and progressive need some memory for their operation. One aspect that distinguishes these from the memory used by LeastVisited , Path, and ProbMap is that the latters' episodic memory requirement grows in general with experience or the history of the agent (with the spatial expanse explored and/or over time), while the former (often control memory) is fixed (constant or near constant).

4 Experiments

We used PYGAME (www.pygame.org) to develop the environments and to visualize. We will make the code available on GitHub. Our experiments involve 3 nested loops: With an outerloop of k_1 trials (*e.g.* 50), we generate initial environments (grids with certain barrier proportion). In an inner loop of k_2 iterations (*e.g.* k_2=20) we generate days: the initial environment is changed somewhat day after day (when change-rate > 0). Finally, within a day, the experiment continues for k_3 steps until the agent, starting from home, reaches food: k_3 depends on agent efficacy. We average k_3 over days and then over the grids, but also report medians and maximums too, see Table 1. For instance max-max refers to taking the maximum of the (k_3) steps over all the days and grids. Each row of the table took seconds to complete on a Mac laptop.

For the composite agents, we used no budget on Greedy, Path, and ProbMap (keep using it until failure), and (initial) budget of 1 on Random and LeastVisited (doubling each time the strategy becomes active again in that day) (see Sect. 3.1). Greedy+Biased means start with Greedy then Biased in round-robin fashion. The agent begins at the center of the grid, and the food is at lower right (unless otherwise specified).This is on 15x15 grids, thus a good path should be about 15 steps long, modulo motion noise and barriers, and the Oracle 's performance reflects that (Table 1).

	mean-mean	med-mean	med-med	max-mean	max-max
Random	1965 ±617	1815	1200	3.5k	30K
Biased	536 ±163	479	329	1056	5.7k
Greedy+Biased	157 ±102	134	24	460	4.4k
LeastVisited	250 ±49	252	198	395	1.5k
Greedy+LeastVisited	87 ±52	73	22	242	1.7k
Path+LeastVisited	224 ±50	227	163	375	2k
ProbMap+LeastVisited	59 ±38	49	23	237	1.3k
Oracle (impractical)	15.5 ±1.1	15.2	14	20.2	51

Table 1: Performance, the number of steps to goal, of a range of strategies, from little or no memory, to extensive memory and planning. ProbMap overall does the best (Oracle gives the minimum in the impractical case of complete knowledge). Experimental settings: 15x15 grids, 50 environments, 20 days each, 0.3 barrier-proportion, 0.1 change-rate, and 0.02 motion noise. Mean-mean is average number of steps over the 20 days and then averaged over the 50 initial environments, while med-mean is the mean of the (50) medians.

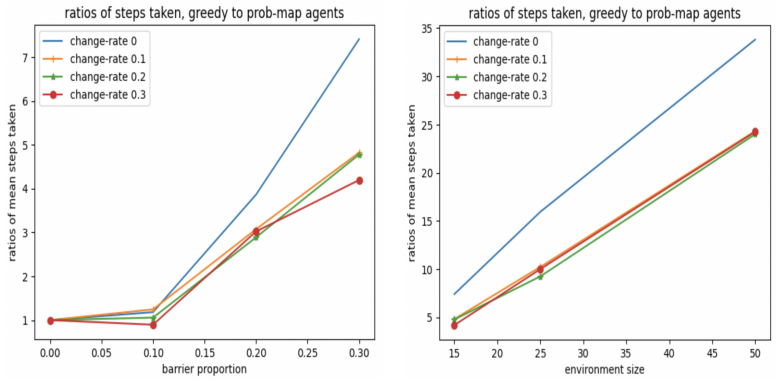


Fig. 5: Efficiency gains, ProbMap+LeastVisited compared to Greedy+Biased, 50 grids, 20 days, (a) When increasing barrier-proportion ratio, for different change rates (on 15x15, 0 motion-noise). (b) When increasing dimensions (from 15x15 to 50x50), keeping barrier proportion at 0.3 (over 30x gains as size increases).

4.1 Memory Helps!

We see that as we add memory, the performances, specially the worst cases, on very bad days (max-max), improve substantially. Biased (with a bit of memory, Sect. 3.3) does substantially better than Random.

The best of memory strategies substantially beats Greedy+Biased, and in particular we get increased robustness. If we ignore the first day or so (the Search days), this gap grows (mean-mean of ProbMap improves to 50). Note that the expected performance of Greedy+Biased can not change from day to

day. The best combination (of smell-direction capability and memory), ProbMap+Greedy+LeastVisited gets a mean of 40 (not shown in the table).

The Path variant trails the Greedy variants in this setting. If we lower the motion noise and change rates to 0, Path gets a mean-mean of 39 beating 70 for Greedy+LeastVisited (and ProbMap gets 18). ProbMap is more robust to change compared to Path. Fig. 5 shows that when grid size or barrier portion difficulties are raised, with low motion-noise, the relative gain of ProbMap compared to Greedy grows, to over 30x (substantial gains, as distance grows, under no noise). Fig. 6 shows that motion noise has the reverse effect, and at some point, Greedy can outperform, and increasing the grid-size can compound this.

Fig. 7 shows that even though the performance of ProbMap+LeastVisited may appear to plateau, the agent needs to keep remembering and learning under (barrier) change to preserve its performance.

4.2 Experiments with a Model-Free NN-Based Agent

We also experimented with DQN [31], a model-free RL technique which performed well on a range of (fully-observable) ATARI games. The agent is given the location, and the surrounding barrier and food information (radius 1, as other agents). On 15x15 grids, and no barriers (and no change in barriers and no motion noise), a learning rate of 0.002 works best (compared to 0.01 and 0.001),[16] and yields a mean of 80 (median 40) for 20 days, and mean of 30 after 50 days, and 20 for 200 days. That is, eventually, the agent learns a good path. However, as we increase barrier proportion, *e.g.* to 0.1, the learning becomes unstable, and on some environments the number of steps even after several days is in the 10s of thousands, to extent that we could not often run such on 20 environments. If we set the change rate to 0.1, the fraction of environments when the agent hangs (has many days with say over 50k steps) goes up. The DQN agent needs both to find the reward, and to pass that information gradually to other cells nearby, and implicitly learn that barriers block. All this, reward propagation, takes times (many steps and days), and if there is change (in goal or barriers), issues of learning stability arise.

4.3 Additional Experiments and Discussion

When the goal location oscillates, eg on the lower right corner on the even days and on the upper right corner on the odd days, the ProbMap+LeastVisited agent remembers the goal pattern, after a handful of days [26] (supporting predictive memory-type in ProbMap enables this capability). We hope to do further longer 'trajectory' experiments, *i.e.* concatenating environments with different characteristics (*e.g.* with different barrier portions, goal locations, or motion noise parameters), and further verify the speed of adaptation.

[16] Other parameters: batch size of 5, gamma of 0.9, eps_start of 0.9, end of 0.05 with decay of 10. We also experimented with changing those parameters.

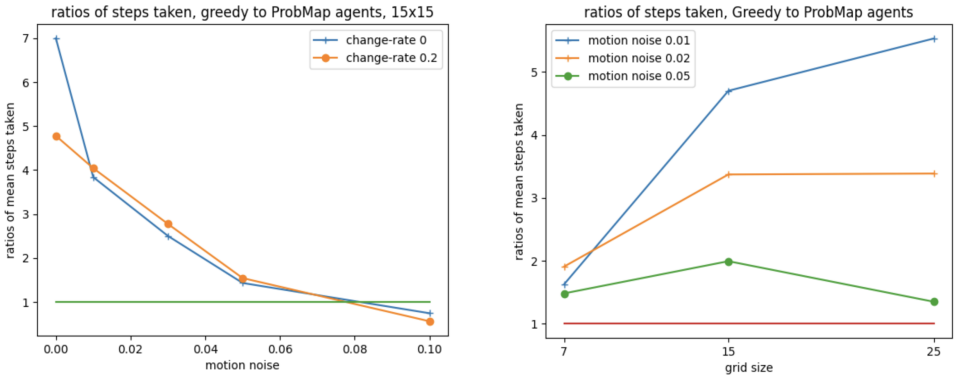


Fig. 6: As motion-noise is increased, ProbMap+LeastVisited loses its advantage over Greedy+Biased. Left: on a fixed 15x15 grid (barrier proportion 0.3), Right: as we increase the grid size (distance to goal). However, with 0 motion-noise, the performance advantage grows with distance (as in Fig. 5).

We note that heading back from food location to home is a similar symmetric problem, and for instance an agent using a map could easily solve that by setting start and goal nodes appropriately when planning.[17] We expect that the basic lessons learned in comparing various strategies generalize. A good general future direction is studying agents that may have to perform multiple tasks.

We have conducted a range of additional experiments: on efficacy of progressive time-budgets (compared to a constant allotment) and the effects and benefits of various parts of ProbMap (such as limiting the memory kept to one day, or making memories deterministic), the number of replannings needed, how memory consumption expands (*e.g.* for ProbMap), and so on. These experiments will be reported in the longer paper [26].

5 Related Work

Our work is related to diverse tracks of research, including autonomous agents, the nature and use of memory (in biological and artificial systems), reinforcement learning (RL), planning, and robotics. We focus on closely related work that was not discussed earlier.

Two broad behaviors or strategies have been identified in computational neuroscience: model-based (*e.g.* map making, and goal oriented) *vs.* habitual instrumental behavior (corresponding to our Path strategy and typical model-free RL): The evidence and the relative strengths and weaknesses are discussed in [10] (such as the simplicity of the habitual *vs.* the flexibility but the computational requirements of goal-oriented model-based behavior). Human memory

[17] For other agents, we may need to make further assumptions. For instance, for the greedy (smell) agent, one could assume that home has its own smell.

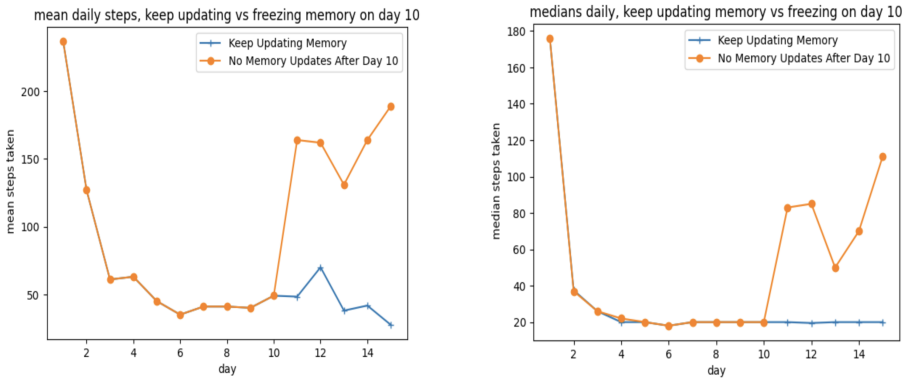


Fig. 7: Continual learning (memory updates) vs freezing memory (*i.e.* stopping updates after day 10). Same setting for Table 1 except for k_1=150 grids. Daily steps, means (left) and medians (right), as a function of day (each point is the mean or median, 150 values, of number of steps taken to food on that day).

representations are complex and are sufficiently flexible to have a diversity of uses, *e.g.* not just for spatial maps, but, for instance, also for the more general cognitive graphs [35, 36]. Learning structured representations, that would find repeated use, may also be foundational for perception [25].

In a partially observed (limited sensing) task [50], the authors study how (predictive) memory and RL techniques could be put together in a perceptually realistic navigation setting, showing promising results that the agent using predictive memory was substantially more successful than plain model-free RL agents (*e.g.* remembering and finding the way to goal when teleported), but the number of episodes (environment steps) remained considerable (*e.g.* 100s of thousand or millions) and generalization ability remains unclear. In a recent study [49], the authors tackle the tricky question of LLM generalization, and develop novel evaluation techniques (*e.g.* querying under small perturbations) for a few tasks and, for a navigation task (in Manhatten, NY), they present evidence that the (transformer) generative network does not learn a systematic ('coherent') map from the sequence training data, *e.g.* when sequences are purtubed from the shortest paths used for training.[18]

Change, in machine learning and RL, continues to be studied and remains a challenge, and techniques in areas such as continual learning, transfer learning, distribution shift, meta learning, lifelong learning, lifelong RL, and open-ended learning in robotics attempt to address the various aspects of the problem [47, 39, 34, 4, 46, 38]. The issue of environment change facing animals was also studied in

[18] The perturbation to action sequences, by which the authors study network performance in one of their experiments, is akin to our random barrier location changes, and to a lesser extent to our motion noise: in their work, the agent (the transformer) is given the alternative move taken.

[55] and the authors propose that animals achieve change detection and change of strategy in part through counter-factual reasoning (as RL does not completely explain the observed speed of change in behavior). See also work on replay [13]. In the area of simultaneous localization and mapping (SLAM) for robotics, richer perception and change is also identified as a future area of research [7].

Recent work, motivated by the possibility of map construction by humans, experiments on subjects as well as performs computational modeling providing evidence that humans build map structures [40] consistent with a Bayesian approach, and furthermore, such maps help planning via partially observed Markov decision processes (POMDPs), in a continual (re)planning fashion. Our study focuses on how repeated change and uncertainty could help or limit the benefits of learning a map, in a simple daily agentic foraging task, and the environment and interaction durations are parameterized (attributes such as grid size and uncertainty characteristics can be changed substantially) and we compare a variety of strategies (pure-sensing or smell-based and path-memory techniques).

6 Summary and Future Directions

The world is large, uncertain, and changing, and an agent has indirect limited access to it. We focused on agents that can remember and learn fast (learning along a lifetime, keeping pace with change), and assumed that certain aspects of the strategies that the agent deploys are hardwired, while other aspects can be continually tuned and learned. We showed that an agent with extensive memory and computing (planning), with appropriate algorithms, can substantially outperform a greedy-smell agent, or extend the greedy's reach, as long as change and uncertainty, in particular in localization, are not too large.

In future work, we hope to reduce the 'hardwired assumptions' we made. For instance, how could an agent come to know what to remember (and how to use it), and how does it 'carve' and granularize its spatial and temporal inputs? Related to this is support for richer perception as well as *open worlds*: for instance learning and using various environmental regularities in a sample efficient and continually adaptable and cumulative manner.

Acknowledgments. Many thanks to the members of our weekly SAIRG group, in particular Christian Martinez, Georgi Georgiev and Gene Lewis, for discussions, pointers, and suggestions, and to the reviewers and BICA organizers.

References

1. P. E. Agre and D. Chapman. Pengi: An implementation of a theory of activity. In *AAAI Conference on Artificial Intelligence*, 1987.
2. P. E. Agre and D. Chapman. What are plans for? *Robotics Auton. Syst.*, 1990.
3. B. Baddeley, P. Graham, P. Husbands, and A. Philippides. A model of ant route navigation driven by scene familiarity. *PLoS*, 2012.
4. M. Botvinick, S. Ritter, J. X. Wang, Z. Kurth-Nelson, C. Blundell, and D. Hassabis. Reinforcement learning, fast and slow. *Trends in cognitive sciences*, 2019.

5. R. A. Brooks. A robust layered control system for a mobile robot. *IEEE J. Robotics Autom.*, 1986.
6. R. A. Brooks. Intelligence without representation. *Artif. Intell.*, 47:139–159, 1991.
7. C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age. *IEEE TRANS. ON ROBOTICS*, 2016.
8. P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. *ArXiv*, 2023.
9. H. Cruse and R. Wehner. No need for a cognitive map: Decentralized memory for insect navigation. *PLoS*, 2011.
10. N. D. Daw. Are we of two minds? *Nature Neuroscience*, 2018.
11. R. D. Estes. The behavior guide to African mammals: Including hoofed mammals, carnivores, primates. 1991.
12. J. Fiedler, E. De Leonibus, and A. Treves. Has the hippocampus really forgotten about space? *Current Opinion in Neurobiology*, 2021.
13. David J. Foster. Replay comes of age. *Annual review of neuroscience*, 2017.
14. S. Franklin. *Artificial Minds*. MIT Press, 1995.
15. S. Franklin and A. C. Graesser. Is it an agent, or just a program?: A taxonomy for autonomous agents. In *ATAL*, 1996.
16. S. Heinze. Navigation: Why flies look to the skies. *eLife*, 2021.
17. S. Heinze, A. Narendra, and A. Cheung. Principles of insect path integration. *Current Biology*, 28:R1043–R1058, 2018.
18. K. Igloi, M. Zaoui, A. Berthoz, and L. Rondi-Reig. Sequential egocentric strategy is acquired as early as allocentric strategy: Parallel acquisition of these two navigation strategies. *Hippocampus*, 2009.
19. T. Ingold. *The Perception of the Environment: Essays on Livelihood, Dwelling, and Skill*. Routledge, 2000. (See Chapter 11 on the temporality of the landscape.).
20. L. P. Kaebling, M. Littman, and A. Moore. Reinforcement learning: a survey. *Artificial Intelligence Research*, 1996.
21. J. Lee, D. Jung, and S. Royer. Stochastic characterization of navigation strategies in an automated variant of the barnes maze. *eLife*, 12, 2024.
22. T. Liang and B. Brinkman. Evolution of innate behavioral strategies through competitive population dynamics. *PLoS*, 2022.
23. M. L. Littman, J. Goldsmith, and M. Mundhenk. The computational complexity of probabilistic planning. *J. Artif. Intell. Res.*, 9, 1998.
24. O. Madani. Tracking changing probabilities via dynamic learners. *arXiv*, 2024.
25. O. Madani. Towards understanding and developing open-ended intelligences for infinite worlds. *To appear in BICA 2025*, 2025.
26. O. Madani, B. Burns, R. Eghbali, and T. Dean. When remembering and planning are worth it: Navigating under change. *ArXiv*, 2025. In Preparation.
27. O. Madani, S. Hanks, and A. Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artif. Intell.*, 2003.
28. A. Matheson, A. Lanz, A. Medina, A. Licata, T. Currier, M. Syed, and K. Nagel. A neural circuit for wind-guided olfactory navigation. *Nature Comm.*, 2022.
29. H. Maturana. Autopoiesis, structural coupling and cognition: A history of these and other notions in the biology of cognition. *Cybern. Hum. Knowing*, 2002.
30. J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 1995.

31. V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. avukcuoglu, D. Wierstra, and M. Riedmiller. Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.
32. M. R. O'Connor. *Wayfinding*. New York : St. Martin's Press, 2019.
33. J. O'Keefe and L. Nadel. *The hippocampus as a cognitive map*. Clarendon Press and Oxford University Press, 1978.
34. G. Parisi, R. Kemker, J. Part, C. Kanan, and S. Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 2018.
35. M. Peer, I. K. Brunec, N. S. Newcombe, and R. A. Epstein. Structuring knowledge with cognitive maps and cognitive graphs. *Trends in Cognitive Sciences*, 2020.
36. Z. Reagh and C. Ranganath. Flexible reuse of cortico-hippocampal representations during encoding and recall of naturalistic events. *Nature Communications*, 2023.
37. C. P. Robert. *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation*. Springer Texts in Statistics, 2001.
38. V. Santucci, P. Oudeyer, A. Barto, and G. Baldassarre. Editorial: Intrinsically motivated open-ended learning in auton. robots. *Front. in Neurorobotics*, 2020.
39. T. Schaul, H. V. Hasselt, J. Modayil, M. White, A. White, P. Bacon, J. Harb, S. Mourad, M. G. Bellemare, and D. Precup. The Barbados 2018 list of open issues in continual learning. *ArXiv*, 2018.
40. S. Sharma, A. Curtis, M. Kryven, J. Tenenbaum, and I. Fiete. Map induction: Compositional spatial submap learning for efficient exploration in novel environments. *ArXiv*, 2021.
41. D. Silver and A. Huang et. al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.
42. B. Sinervo. *Optimal Foraging Theory: Constraints and Cognitive Processes*, chapter 6. University of California, Santa Cruz, 1997.
43. K. L. Stachenfeld, M. M. Botvinick, and S. J. Gershman. The hippocampus as a predictive map. *Nature Neuroscience*, 20:1643–1653, 2017.
44. J. M. Starck and R. E. Ricklefs. Avian growth and development: evolution within the altricial-precocial spectrum. 1998.
45. R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.
46. R. S. Sutton, M. Bowling, and P. M. Pilarski. The Alberta plan for AI research. *ArXiv*, 2022.
47. S. Thrun and T. M. Mitchell. Lifelong robot learning. *Robotics Auton. Syst.*, 1993.
48. E. C. Tolman. Cognitive maps in rats and men. *Psychological Review*, 1948.
49. K. Vafa, J. Chen, A. Rambachan, J. Kleinberg, and S. Mullainathan. Evaluating the world model implicit in a generative model. *NeuroIPS*, 2024.
50. G. Wayne, C. Hung, D. Amos, M. Mirza, and A. Ahuja et. al. Unsupervised predictive memory in a goal-directed agent. *arXiv*, 2018.
51. R. Wehner. The ant's celestial compass system: spectral and polarization channels. In *Orientation and communication in arthropods*, pages 145–185. Springer, 1997.
52. K. Wijnen, L. Genzel, and J. van der Meij. Rodent maze studies: from following simple rules to complex map learning. *Brain Structure & Function*, 2024.
53. M. Yaghoubi, A. Nieto-Posadas, C. Mosser, T. Gisiger, É. Wilson, S. Williams, and M. P. Brandon. Predictive coding of reward in the hippocampus. *bioRxiv*, 2024.
54. V. Zaburdaev, S. Denisov, and J. Klafter. Lévy walks. *Rev. Modern Physics*, 2014.
55. Y. Zhang, J. Paik, and P. Pirolli. Reinforcement learning and counterfactual reasoning explain adaptive behavior in a changing environment. *Topics in cognitive science*, 2015.