Lecture 13: Simulation and planning with mental models

Samuel Gershman

Harvard University

Roadmap

- Pinnacle of flexibility: mental models that support simulation and planning. This lecture discusses evidence for mental models in the brain.

Roadmap

- Pinnacle of flexibility: mental models that support simulation and planning. This lecture discusses evidence for mental models in the brain.
- We formalize the function of mental models in terms of model-based solutions to sequential decision problems.

Roadmap

- Pinnacle of flexibility: mental models that support simulation and planning. This lecture discusses evidence for mental models in the brain.
- We formalize the function of mental models in terms of model-based solutions to sequential decision problems.
- Some model-based algorithms use offline simulation to provide synthetic data for training model-free algorithms.

Roadmap

- Pinnacle of flexibility: mental models that support simulation and planning. This lecture discusses evidence for mental models in the brain.
- We formalize the function of mental models in terms of model-based solutions to sequential decision problems.
- Some model-based algorithms use offline simulation to provide synthetic data for training model-free algorithms.
- Other model-based algorithms use online simulation to evaluate different courses of action at decision time.

Roadmap

- Pinnacle of flexibility: mental models that support simulation and planning. This lecture discusses evidence for mental models in the brain.
- We formalize the function of mental models in terms of model-based solutions to sequential decision problems.
- Some model-based algorithms use offline simulation to provide synthetic data for training model-free algorithms.
- Other model-based algorithms use online simulation to evaluate different courses of action at decision time.
- Unifying these two approaches is the idea that the brain can imagine answers to "what if?" questions, liberating itself from the prison of pure experience.

Spatial navigation

- We will focus on spatial navigation, which provides the most well-studied examples, and which carries ecological significance for many species.

Spatial navigation

- We will focus on spatial navigation, which provides the most well-studied examples, and which carries ecological significance for many species.
- Even animals with relatively simple nervous systems, such as ants, are capable of tracking their spatial location and using this information for charting a path toward specific landmarks.

Spatial navigation

- We will focus on spatial navigation, which provides the most well-studied examples, and which carries ecological significance for many species.
- Even animals with relatively simple nervous systems, such as ants, are capable of tracking their spatial location and using this information for charting a path toward specific landmarks.
- In some cases, animals use their knowledge to plan detour and shortcut routes.

Spatial navigation

- We will focus on spatial navigation, which provides the most well-studied examples, and which carries ecological significance for many species.
- Even animals with relatively simple nervous systems, such as ants, are capable of tracking their spatial location and using this information for charting a path toward specific landmarks.
- In some cases, animals use their knowledge to plan detour and shortcut routes.
- We will consider what kinds of neural mechanisms could support these computations.

Spatial navigation in ants

- The desert ant *Cataglyphis*, despite its small brain (less than 1 million neurons), is capable of incredible navigational feats.

Spatial navigation in ants

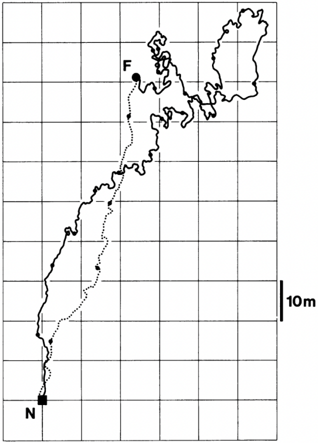
- The desert ant *Cataglyphis*, despite its small brain (less than 1 million neurons), is capable of incredible navigational feats.
- While foraging, the ant follows a circuitous path until it finds food.

Spatial navigation in ants

- The desert ant *Cataglyphis*, despite its small brain (less than 1 million neurons), is capable of incredible navigational feats.
- While foraging, the ant follows a circuitous path until it finds food.
- Rather than retracing its steps, it orients toward its nest and follows a more or less straight path home—a hole 1 mm in diameter, which might be over 100 m away.

Spatial navigation in ants

Foraging (solid line) and return (dotted line) trajectories of the desert ant *Cataglyphsis*. F = food, N = nest.



[Muller & Wehner 1988]

Spatial navigation in ants

- If the ant is displaced far from its nest after finding food (so that it can't rely on visual landmarks), it will orient in the direction where its nest *would have been* had it not been displaced, and follows a straight-line path that terminates close to the counterfactual location of the nest [Wehner & Srinivasan 1981].

Spatial navigation in ants

- If the ant is displaced far from its nest after finding food (so that it can't rely on visual landmarks), it will orient in the direction where its nest *would have been* had it not been displaced, and follows a straight-line path that terminates close to the counterfactual location of the nest [Wehner & Srinivasan 1981].
- Ants trained with consistent food locations follow straight-line outbound paths from the nest back to those locations [Collett et al 1999]. These "food vector" memories appear to last the ant's entire lifetime.

Path integration in ants

- Cognitive map hypothesis: ants keep track of their 2D location relative to the nest, and then use this to chart a "homing" vector back to the nest.

Path integration in ants

- Cognitive map hypothesis: ants keep track of their 2D location relative to the nest, and then use this to chart a "homing" vector back to the nest.
- The hypothesized algorithm for keeping track of position—*path integration*—is essentially identical to the *dead reckoning* algorithm used by human sailors for thousands of years.

Path integration in ants

- Cognitive map hypothesis: ants keep track of their 2D location relative to the nest, and then use this to chart a "homing" vector back to the nest.
- The hypothesized algorithm for keeping track of position—*path integration*—is essentially identical to the *dead reckoning* algorithm used by human sailors for thousands of years.
- Core idea: position is the integral of velocity over time, so that a position vector can be computed by adding up movements over time.

Path integration as model-based reasoning

- Path integration is an elementary form of model-based reasoning: what happens to my location when I move?

Path integration as model-based reasoning

- Path integration is an elementary form of model-based reasoning: what happens to my location when I move?
- We can identify this with a representation of the transition function: state = location, action = movement vector.

Path integration as model-based reasoning

- Path integration is an elementary form of model-based reasoning: what happens to my location when I move?
- We can identify this with a representation of the transition function: state = location, action = movement vector.
- Access to a path integration system can be harnessed by more powerful algorithms as a forward simulator even in the absence of overt movement (i.e., a form of imagination), opening the door to online planning and offline learning abilities.

Path integration defined formally

- Let $s_t \in \mathbb{R}^2$ denote the agent's 2D spatial position at time t. Action $a_t \in \mathbb{R}^2$ is a velocity vector, specifying the instantaneous change in position: $\dot{s}_t = a_t$.

Path integration defined formally

- Let $s_t \in \mathbb{R}^2$ denote the agent's 2D spatial position at time t. Action $a_t \in \mathbb{R}^2$ is a velocity vector, specifying the instantaneous change in position: $\dot{s}_t = a_t$.
- If the position variable is initialized to $[0, 0]$ at time 0, the path integral of velocity yields position in the reference frame centered at the initial position:

$$s_t = \int_0^t a_{\tilde{t}} d\tilde{t}$$

Path integration defined formally

- Let $s_t \in \mathbb{R}^2$ denote the agent's 2D spatial position at time t. Action $a_t \in \mathbb{R}^2$ is a velocity vector, specifying the instantaneous change in position: $\dot{s}_t = a_t$.
- If the position variable is initialized to $[0, 0]$ at time 0, the path integral of velocity yields position in the reference frame centered at the initial position:

$$s_t = \int_0^t a_{\tilde{t}} d\tilde{t}$$

- For the foraging ant, the initial position is the nest; the integrated position is represented in nest-centered coordinates.

Homing and memory

- Once food is found, the ant can chart a path home by aligning its movement direction with the homing vector, $-s_t$.

Homing and memory

- Once food is found, the ant can chart a path home by aligning its movement direction with the homing vector, $-s_t$.
- Once the homing vector is close to $[0, 0]$, the ant knows that it's in the vicinity of the nest (up to noise in the integrator).

Homing and memory

- Once food is found, the ant can chart a path home by aligning its movement direction with the homing vector, $-s_t$.
- Once the homing vector is close to $[0, 0]$, the ant knows that it's in the vicinity of the nest (up to noise in the integrator).
- Food location can be stored in memory, so that the ant can return to it from the nest by aligning its movement direction with the vector pointing at the stored location.

Continuous attractor network implementation

- Recurrently connected neurons with population activity x, recurrent weight matrix W. continuous-time firing rate dynamics:

  $$\tau \dot{x} = -x + \phi(Wx + Ga)$$

  where $\phi(\cdot)$ is a non-linearity, and a is the velocity drive acting through the feedforward matrix G, which implicitly encodes the preferred movement direction θ_i for each neuron i.

Continuous attractor network implementation

- Recurrently connected neurons with population activity x, recurrent weight matrix W. continuous-time firing rate dynamics:

$$\tau \dot{x} = -x + \phi(Wx + Ga)$$

  where $\phi(\cdot)$ is a non-linearity, and a is the velocity drive acting through the feedforward matrix G, which implicitly encodes the preferred movement direction θ_i for each neuron i.
- Each neuron also has a preferred position $\bar{s}_i$, which governs the "difference of Gaussians" connectivity profile:

$$W_{ij} = K(\bar{s}_i - \bar{s}_j - g_j), \qquad K(s) = \exp[-\alpha \|s\|^2] - \exp[-\beta \|s\|^2]$$

  where α and β control excitation vs. inhibition.

Continuous attractor network implementation

- Recurrently connected neurons with population activity x, recurrent weight matrix W. continuous-time firing rate dynamics:

  $$\tau \dot{x} = -x + \phi(Wx + Ga)$$

  where $\phi(\cdot)$ is a non-linearity, and a is the velocity drive acting through the feedforward matrix G, which implicitly encodes the preferred movement direction θ_i for each neuron i.
- Each neuron also has a preferred position $\bar{s}_i$, which governs the "difference of Gaussians" connectivity profile:

  $$W_{ij} = K(\bar{s}_i - \bar{s}_j - g_j), \qquad K(s) = \exp[-\alpha \|s\|^2] - \exp[-\beta \|s\|^2]$$

  where α and β control excitation vs. inhibition.
- Neurons that prefer nearby positions excite one another. Inhibitory term produces surround suppression. Shift term g_j skews activity in the direction that the agent is moving.

Continuous attractor network implementation

- Network will form stable "bumps" of activity at particular positions.

Continuous attractor network implementation

- Network will form stable “bumps” of activity at particular positions.
- Bumps will be restored following small perturbations of the activity state—the bumps are *attractors*.

Continuous attractor network implementation

- Network will form stable “bumps” of activity at particular positions.
- Bumps will be restored following small perturbations of the activity state—the bumps are *attractors*.
- The set of attractors forms a continuous manifold in position space: smooth changes in position generate locally Euclidean changes in neural activity.

Continuous attractor network implementation

- Activity organized on a sheet, where neurons are placed topographically based on their preferred location. Each patch of the sheet contains neurons covering all preferred directions.

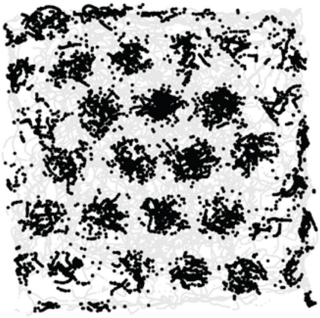
Continuous attractor network implementation

- Activity organized on a sheet, where neurons are placed topographically based on their preferred location. Each patch of the sheet contains neurons covering all preferred directions.
- When animal is stationary at a particular location, a periodic grid of neurons is activated on the sheet, similar to “grid cells” in the entorhinal cortex that fire periodically across space.

Continuous attractor network implementation

- Activity organized on a sheet, where neurons are placed topographically based on their preferred location. Each patch of the sheet contains neurons covering all preferred directions.
- When animal is stationary at a particular location, a periodic grid of neurons is activated on the sheet, similar to “grid cells” in the entorhinal cortex that fire periodically across space.
- The periodicity comes from the center-surround interactions between nearby neurons on the sheet.

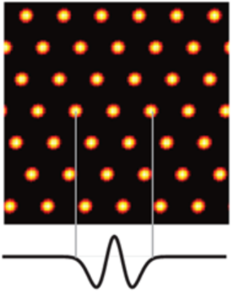
Entorhinal grid cells



[Moser & Moser 2013]

Neural activity organized into a grid pattern

Activity on the neural sheet. The center-surround kernel, $K(x)$, is shown on the bottom.



[Burak & Fiete 2009]

Continuous attractor network implementation

- Non-zero velocity drive causes the activity bump to translate on the sheet.

Continuous attractor network implementation

- Non-zero velocity drive causes the activity bump to translate on the sheet.
- For grid cells, this means that individual neurons fire periodically as an animal traverses space, while maintaining their phase relationships.

Continuous attractor network implementation

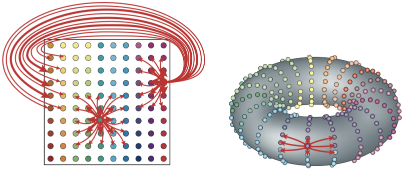
- Non-zero velocity drive causes the activity bump to translate on the sheet.
- For grid cells, this means that individual neurons fire periodically as an animal traverses space, while maintaining their phase relationships.
- Position can be decoded linearly from phase up to the period of the grid. Geometrically, this corresponds to identifying a position on a ring wrapping around the torus representation of the sheet.

Continuous attractor network implementation

- Non-zero velocity drive causes the activity bump to translate on the sheet.
- For grid cells, this means that individual neurons fire periodically as an animal traverses space, while maintaining their phase relationships.
- Position can be decoded linearly from phase up to the period of the grid. Geometrically, this corresponds to identifying a position on a ring wrapping around the torus representation of the sheet.
- Using this readout, total positional error accumulated over 20 minutes (covering around 260 meters) is less than 15 centimeters.

Torus topology

Neural connectivity with periodic boundary conditions. (Left) Neural connectivity wraps around the neural sheet. (Right) The connectivity is topologically equivalent to a torus.



[McNaughton et al 2006]

Why a periodic representation of space?

- At first glance, it seems strange to use a periodic representation of space to keep track of position. If position can only be identified up to the grid period, this representation will be useless for any environments that are larger than the grid period.

Why a periodic representation of space?

- At first glance, it seems strange to use a periodic representation of space to keep track of position. If position can only be identified up to the grid period, this representation will be useless for any environments that are larger than the grid period.
- Critically, the entorhinal cortex contains multiple modules, each with a different period, giving rise to a multi-scale representation of space.

Why a periodic representation of space?

- At first glance, it seems strange to use a periodic representation of space to keep track of position. If position can only be identified up to the grid period, this representation will be useless for any environments that are larger than the grid period.
- Critically, the entorhinal cortex contains multiple modules, each with a different period, giving rise to a multi-scale representation of space.
- By combining the modules, positional information can be tracked over very large spaces [Fiete et al 2008].

Shortcuts and detours

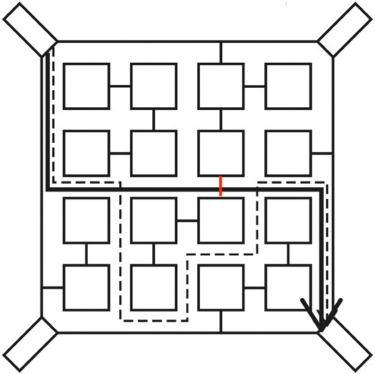
- A spatial model enables more than tracking position and computing homing vectors. An agent can also plan shortcuts and detours, using the same path integration mechanism in its imagination rather than during actual movement.

Shortcuts and detours

- A spatial model enables more than tracking position and computing homing vectors. An agent can also plan shortcuts and detours, using the same path integration mechanism in its imagination rather than during actual movement.
- This allows agents to exploit shortcuts and navigate detours.

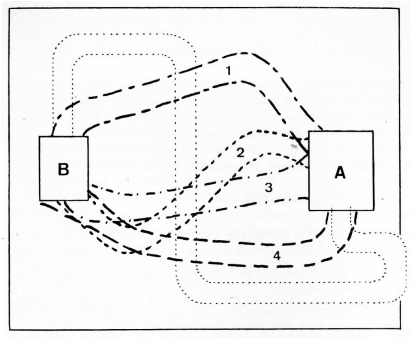
Shortcuts in a lattice maze

Red line shows the wall that was removed in the shortcut test. The dashed black line shows the path rats took during training; the solid black line shows the path rats took during the shortcut test.



[Sato et al 2018]

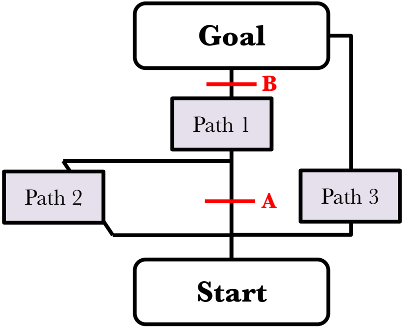
Shortcuts in burrowing behavior



[Zanforlin & Poli 1970]

Detour behavior

When a path is blocked, rats tend to take the shortest path that gets them to the goal.



[Tolman & Honzik 1930]

Mental simulation

- It's not enough to compute the direction of the goal, because in many of these cases animals have to move *away* from the goal in order to take the shortest path.

Mental simulation

- It's not enough to compute the direction of the goal, because in many of these cases animals have to move *away* from the goal in order to take the shortest path.
- What's needed is the ability to look ahead in a cognitive map, until a sufficiently short path is found.

Mental simulation

- It's not enough to compute the direction of the goal, because in many of these cases animals have to move *away* from the goal in order to take the shortest path.
- What's needed is the ability to look ahead in a cognitive map, until a sufficiently short path is found.
- **Vicarious trial and error**: rats often pause at choice points (intersections) in a maze, looking in each direction before moving again.

Mental simulation

- It's not enough to compute the direction of the goal, because in many of these cases animals have to move *away* from the goal in order to take the shortest path.
- What's needed is the ability to look ahead in a cognitive map, until a sufficiently short path is found.
- **Vicarious trial and error**: rats often pause at choice points (intersections) in a maze, looking in each direction before moving again.
- Place cells sweep ahead of the rat's actual position at choice points [Johnson & Redish 2007].

Place cell activity during vicarious trial and error

Decoded spatial position from hippocampal ensembles at a choice point. Note that the rat is not moving; thus, the decoded position is imagined.

[Johnson & Redish 2007]

Mental simulation

- A path integration system could implement forward sweeps by applying velocity drive to the integrator.

Mental simulation

- A path integration system could implement forward sweeps by applying velocity drive to the integrator.
- In a cluttered environment, the velocity drive would need to point not directly at the goal but rather at locally accessible subgoals.

Mental simulation

- A path integration system could implement forward sweeps by applying velocity drive to the integrator.
- In a cluttered environment, the velocity drive would need to point not directly at the goal but rather at locally accessible subgoals.
- Hippocampal sweeps appear to be chunked based on significant landmarks or choice points.

Mental simulation

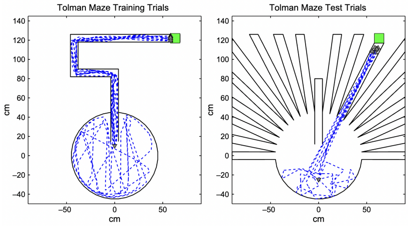
- How this might work: head direction cells (neurons tuned to specific head directions) supply the directional component of velocity drive [Erdem & Hasselmo 2012].

Mental simulation

- How this might work: head direction cells (neurons tuned to specific head directions) supply the directional component of velocity drive [Erdem & Hasselmo 2012].
- Resulting sweeps produce locally linear look-ahead; by chaining these sweeps together, the simulation can eventually reach a goal.

Linear look-ahead

Dashed lines show simulated trajectories.



[Erdem & Hasselmo 2012]

Mental simulation

- How do the head direction cells know what direction to activate?

Mental simulation

- How do the head direction cells know what direction to activate?
- One possibility: head direction cells are guided by a representation (possibly in prefrontal cortex) of reward signals at particular locations.

Mental simulation

- How do the head direction cells know what direction to activate?
- One possibility: head direction cells are guided by a representation (possibly in prefrontal cortex) of reward signals at particular locations.
- This signal diffuses through the cognitive map, such that a short linear look-ahead can contact it and subsequently project further look-aheads in that direction.

Mental simulation

- How do the head direction cells know what direction to activate?
- One possibility: head direction cells are guided by a representation (possibly in prefrontal cortex) of reward signals at particular locations.
- This signal diffuses through the cognitive map, such that a short linear look-ahead can contact it and subsequently project further look-aheads in that direction.
- This provides the key mechanism for shortcut discovery, since the diffusion will tend to concentrate along shorter paths.

Connection to modern planning algorithms

- The linear look-ahead $+$ reward diffusion model has some intriguing similarities with modern planning algorithms, which have been instrumental to the design of high-performance AI systems.

Connection to modern planning algorithms

- The linear look-ahead + reward diffusion model has some intriguing similarities with modern planning algorithms, which have been instrumental to the design of high-performance AI systems.
- AlphaGo [Silver et al 2016] uses a value function approximator to guide look-ahead. The values are estimated with a form of TD learning.

Connection to modern planning algorithms

- The linear look-ahead + reward diffusion model has some intriguing similarities with modern planning algorithms, which have been instrumental to the design of high-performance AI systems.
- AlphaGo [Silver et al 2016] uses a value function approximator to guide look-ahead. The values are estimated with a form of TD learning.
- In an environment with a single goal, this produces a diffusion-like propagation of information.

Connection to modern planning algorithms

- The linear look-ahead + reward diffusion model has some intriguing similarities with modern planning algorithms, which have been instrumental to the design of high-performance AI systems.
- AlphaGo [Silver et al 2016] uses a value function approximator to guide look-ahead. The values are estimated with a form of TD learning.
- In an environment with a single goal, this produces a diffusion-like propagation of information.
- Important difference: general planning algorithms do not typically assume that the state space is Euclidean (an assumption necessary for linear look-ahead).

Mental navigation in non-Euclidean state spaces

- A Euclidean state space is geometrically self-consistent in the sense that if you add together a sequence of translations and end up where you started, then the sum of the translations must equal 0—a *loop closure* property.

Mental navigation in non-Euclidean state spaces

- A Euclidean state space is geometrically self-consistent in the sense that if you add together a sequence of translations and end up where you started, then the sum of the translations must equal 0—a *loop closure* property.
- This is the basis of the desert ant's homing behavior: following the homing vector ensures that its path integrator returns to 0, coinciding with arrival at its nest.

Mental navigation in non-Euclidean state spaces

- A Euclidean state space is geometrically self-consistent in the sense that if you add together a sequence of translations and end up where you started, then the sum of the translations must equal 0—a *loop closure* property.
- This is the basis of the desert ant's homing behavior: following the homing vector ensures that its path integrator returns to 0, coinciding with arrival at its nest.
- Non-Euclidean state spaces don't necessarily obey loop closure. For example, the Earth is only locally Euclidean (the ground looks flat over short distances); over longer distances, the curvature of the Earth starts to matter, and the homing vector will not necessarily get the ant back to its nest, even if the path integrator returns to 0.

Mental navigation in non-Euclidean state spaces

- The brain needs to plan in many state spaces that are not obviously Euclidean.

Mental navigation in non-Euclidean state spaces

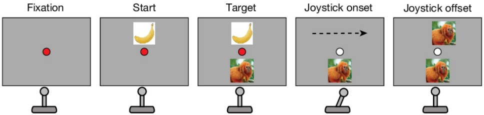
- The brain needs to plan in many state spaces that are not obviously Euclidean.
- Can we represent these spaces in a way that is at least approximately Euclidean? If so, then we can harness the path integration capability of the entorhinal cortex for planning.

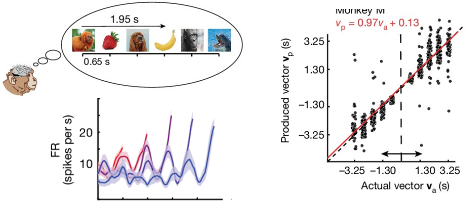
Mental navigation in non-Euclidean state spaces

- The brain needs to plan in many state spaces that are not obviously Euclidean.
- Can we represent these spaces in a way that is at least approximately Euclidean? If so, then we can harness the path integration capability of the entorhinal cortex for planning.
- Some evidence: monkeys trained to move through an abstract space of landmarks using a joystick. When they moved the joystick, movement through the space was invisible (they couldn't see the sequence of landmarks intervening between the start and end points); nonetheless, neurons in entorhinal cortex exhibited periodic firing patterns, with bumps that aligned to the expected timing of landmark arrivals.

Signatures of mental navigation in an abstract space

Monkeys used a joystick to navigate between visual "landmarks" (images) arranged on a 1-D line, illustrated in the schematic. Monkeys learned to produce the correct translation vector that would bring them to a target landmark. As the translation was unfolding invisibly, entorhinal cells fired periodically, with a period matched to the inter-landmark interval.





[Neupane et al 2024]

Mental navigation in non-Euclidean state spaces

- The brain needs to plan in many state spaces that are not obviously Euclidean.

Mental navigation in non-Euclidean state spaces

- The brain needs to plan in many state spaces that are not obviously Euclidean.
- Can we represent these spaces in a way that is at least approximately Euclidean? If so, then we can harness the path integration capability of the entorhinal cortex for planning.

Mental navigation in non-Euclidean state spaces

- The brain needs to plan in many state spaces that are not obviously Euclidean.
- Can we represent these spaces in a way that is at least approximately Euclidean? If so, then we can harness the path integration capability of the entorhinal cortex for planning.
- Some evidence: monkeys trained to move through an abstract space of landmarks using a joystick. When they moved the joystick, movement through the space was invisible (they couldn't see the sequence of landmarks intervening between the start and end points); nonetheless, neurons in entorhinal cortex exhibited periodic firing patterns, with bumps that aligned to the expected timing of landmark arrivals.

Mental navigation in non-Euclidean state spaces

- Periodic structure of grid cells in entorhinal cortex extend beyond spatial navigation, reflecting a common underlying principle.

Mental navigation in non-Euclidean state spaces

- Periodic structure of grid cells in entorhinal cortex extend beyond spatial navigation, reflecting a common underlying principle.
- One way to formalize this principle is to optimize an RNN that transforms neural state representations by running the dynamics forward.

Mental navigation in non-Euclidean state spaces

- Periodic structure of grid cells in entorhinal cortex extend beyond spatial navigation, reflecting a common underlying principle.
- One way to formalize this principle is to optimize an RNN that transforms neural state representations by running the dynamics forward.
- By mapping the state representations to observations, the RNN can match its predictions with these observations.

Mental navigation in non-Euclidean state spaces

- Periodic structure of grid cells in entorhinal cortex extend beyond spatial navigation, reflecting a common underlying principle.
- One way to formalize this principle is to optimize an RNN that transforms neural state representations by running the dynamics forward.
- By mapping the state representations to observations, the RNN can match its predictions with these observations.
- In open-field environments this produces grid-like periodic representations.

Mental navigation in non-Euclidean state spaces

- Periodic structure of grid cells in entorhinal cortex extend beyond spatial navigation, reflecting a common underlying principle.
- One way to formalize this principle is to optimize an RNN that transforms neural state representations by running the dynamics forward.
- By mapping the state representations to observations, the RNN can match its predictions with these observations.
- In open-field environments this produces grid-like periodic representations.
- These representations form an abstract structural code: they can be reused in any task that shares the same underlying spatial structure.

General planning algorithms

- What do you do if the state space can't be treated as approximately Euclidean?

General planning algorithms

- What do you do if the state space can't be treated as approximately Euclidean?
- Dynamic programming (harness the Bellman equation), tree search (harness the forward simulator).

Tree search

- Modern large-scale planning algorithms (like AlphaGo) use "rollouts" of simulated state-action sequences initiated from the current state.

Tree search

- Modern large-scale planning algorithms (like AlphaGo) use "rollouts" of simulated state-action sequences initiated from the current state.
- These rollouts can be understood as search through the decision tree rooted at the current state, hence the name *tree search* for this general family of algorithms.

Tree search

- Modern large-scale planning algorithms (like AlphaGo) use "rollouts" of simulated state-action sequences initiated from the current state.
- These rollouts can be understood as search through the decision tree rooted at the current state, hence the name *tree search* for this general family of algorithms.
- Tree search algorithms estimate the value function locally, so they don't suffer from unfavorable scaling with the size of the state space.

Tree search

- Modern large-scale planning algorithms (like AlphaGo) use "rollouts" of simulated state-action sequences initiated from the current state.
- These rollouts can be understood as search through the decision tree rooted at the current state, hence the name *tree search* for this general family of algorithms.
- Tree search algorithms estimate the value function locally, so they don't suffer from unfavorable scaling with the size of the state space.
- This locality means that action policies are not guaranteed to be globally consistent over the entire state space.

Tree search

- The trick to making tree search algorithms efficient is designing the rollout policy to selectively search promising parts of the decision tree.

Tree search

- The trick to making tree search algorithms efficient is designing the rollout policy to selectively search promising parts of the decision tree.
- A simple heuristic is to myopically prune sub-trees whenever an unfavorable outcome is encountered. Humans relinquish high-payoff paths that require traversing a large loss [Huys et al 2012].

Tree search

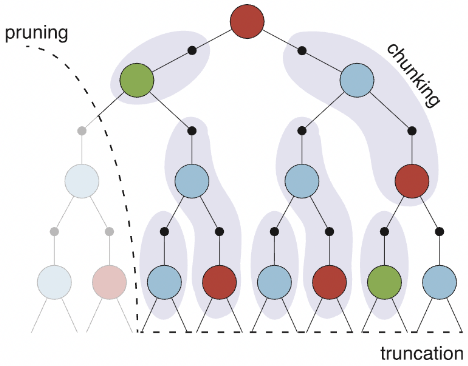
- The trick to making tree search algorithms efficient is designing the rollout policy to selectively search promising parts of the decision tree.
- A simple heuristic is to myopically prune sub-trees whenever an unfavorable outcome is encountered. Humans relinquish high-payoff paths that require traversing a large loss [Huys et al 2012].
- More sophisticated, non-myopic heuristic: use learned value function to guide the rollouts, a key design feature of systems like AlphaGo.

Tree search

- The trick to making tree search algorithms efficient is designing the rollout policy to selectively search promising parts of the decision tree.
- A simple heuristic is to myopically prune sub-trees whenever an unfavorable outcome is encountered. Humans relinquish high-payoff paths that require traversing a large loss [Huys et al 2012].
- More sophisticated, non-myopic heuristic: use learned value function to guide the rollouts, a key design feature of systems like AlphaGo.
- Evidence from experiments in which human subjects are asked to externalize their rollouts (by trying out paths before committing to one) suggests that they can adopt non-myopic heuristics, taking into account both the long-term value and their own uncertainty when selecting rollouts [Fan et al 2025].

Tree search heuristics

Each node represents a state, and each edge represents an action.



[Mattar & Lengyel 2022]

Rollouts in the hippocampus

- Hippocampal sweeps interpreted as rollouts [Jensen et al 2024].

Rollouts in the hippocampus

- Hippocampal sweeps interpreted as rollouts [Jensen et al 2024].
- Sequences tend to (i) avoid passing through walls; (ii) reach the goal; (iii) predict the next physical movement specifically when the sequence reaches the goal; and (iv) increase the rate at which the goal is reached over multiple sequences.

Study question

What do you expect would happen to sequential choice behavior if the hippocampus is lesioned?

Summary

- Model-based reasoning is often thought to be the pinnacle of cognition, underlying our most impressive feats of flexibility.

Summary

- Model-based reasoning is often thought to be the pinnacle of cognition, underlying our most impressive feats of flexibility.
- Elements of an internal world model can be seen not only in humans but even in much simpler creatures like ants and wasps.

Summary

- Model-based reasoning is often thought to be the pinnacle of cognition, underlying our most impressive feats of flexibility.
- Elements of an internal world model can be seen not only in humans but even in much simpler creatures like ants and wasps.
- Several ways an internal model can be used: homing behavior (in spatial navigation tasks), goal-directed planning, mental simulation.

Summary

- Model-based reasoning is often thought to be the pinnacle of cognition, underlying our most impressive feats of flexibility.
- Elements of an internal world model can be seen not only in humans but even in much simpler creatures like ants and wasps.
- Several ways an internal model can be used: homing behavior (in spatial navigation tasks), goal-directed planning, mental simulation.
- Path integration is an elementary form of model-based reasoning; can be implemented in a continuous attractor network; can be repurposed for conceptual state spaces with approximately Euclidean topology.

Summary

- Model-based reasoning is often thought to be the pinnacle of cognition, underlying our most impressive feats of flexibility.
- Elements of an internal world model can be seen not only in humans but even in much simpler creatures like ants and wasps.
- Several ways an internal model can be used: homing behavior (in spatial navigation tasks), goal-directed planning, mental simulation.
- Path integration is an elementary form of model-based reasoning; can be implemented in a continuous attractor network; can be repurposed for conceptual state spaces with approximately Euclidean topology.
- Brain applies general planning algorithms, like dynamic programming and tree search, to non-Euclidean state spaces.