Artificial Enactive Inference in Three-Dimensional World

Olivier L. Georgeon, David Lurie, Paul Robertson

Artificial Enactive Inference in Three-Dimensional World

Olivier L. Georgeon[a], David Lurie[b], Paul Robertson[c]

[a] *UR CONFLUENCE: Sciences et Humanites (EA 1598), UCLy, Lyon Catholic Universite, France*
[b] *Paris-Dauphine University - PSL, NyxAir, France*
[c] *DOLL Labs, Lexington, MA, USA*

Abstract

The theory of Enactive Inference was proposed by Karl Friston and his colleagues to explain how the brain infers knowledge about the world through the subject's interactive experiences. Sensorimotor states induce perturbations in neural activity, and the brain infers hypothetical causes in the world that may explain these perturbations. This article aims to reconcile this neuroscience theory with computer science and artificial-intelligence theories wherein artificial agents receive input data derived from the environment's state and infer internal data structures used to guide decisions. Two critical challenges arise in both the agent's active role and the inference algorithm's scalability as the environment's complexity increases. To address these challenges, we formalize artificial enactive inference through a new Spatial Enactive Markov Decision Process (SEMDP) model. This model rests on low-level control loops enacted in a three-dimensional Euclidean space containing objects. Based on the SEMDP, we present a proof-of-concept cognitive architecture and an experiment to demonstrate the transcription of the theory of enactive inference into the domain of artificial intelligence and robotics.

Keywords: Enaction, active inference, constructivist learning, Dynamic Bayesian Networks, POMDPs, robotics

1. Introduction

Maxwell Ramstead, Michael Kirchhoff, and Karl Friston [1] proposed the concept of *enactive inference* when they gave an "enactive interpretation" of the concept of *active inference* previously proposed by their research group

[e.g., 2]. Active inference involves a kind of *Bayesian inference* which is a statistical method to assess a possible underlying *"model"* (or *hypothesis*) of the features of the world that may *"cause"* sensory signals. They stress that the theory of active inference "emphasizes the tight coupling and circular causality between perception and action" and that "it concerns the active, selective sampling of the world by an embodied agent" [1, p. 226]. Following these arguments, they coined the expression *enactive inference* to refer to the process by which an autonomous agent "simultaneously performs inference and control" in a "causally circular embrace" in which "the actual causes of sensory input depend on action, while action depends on inference."

The adjective *enactive* refers to the broader scope of the theory of enaction, which, according to Daniel Hutto and Erik Myin [3] can be divided in three major strands: *autopoietic*, *sensorimotor*, and *radical*. Autopoietic enactivism [4] focuses on grounding cognition in biological dynamics of a living system. Sensorimotor enactivism [5] focuses on explaining the character of perceptual experience in terms of knowledge or "mastery" of sensorimotor contingencies. Radical enactivism [3] focuses primarily on the anti-representational nature of enactivism. As will be evident in the foregoing, this paper falls within the sensorimotor enactivism strand. Indeed, enactive inference focuses on inferring knowledge from sensorimotor interaction. While enactive inference theory proposes an explanation of the agent's drives through the *free energy principle*, it is not mainly focused on elaborating the agent's goals and motivations, as the autopoietic strand is. While it rejects "structural representationalism", it accepts "internal models" which distinguishes it from the radical strand.

The enactive inference theory proposes that the agent infers *generative* and *recognition* models. The generative model is a statistical model of how sensory signals are generated. The recognition model is a statistical model of the hypothetical causes of the sensory signals; that is, the agent's "best guess" about features of the world that it cannot observe directly. Both models work in tandem: the generative model harnesses prior beliefs at the beginning of the interaction cycle and produces an anticipation of expected sensory outcomes. The recognition model harnesses posterior beliefs at the end of the interaction cycle and produces hypotheses that may explain the sensory signals.

In computer science, a handful of authors have studied the implementation of sensorimotor enactivism in artificial agents. For example, Alexander Maye and Andreas Engel [6] used Markov models to describe sequences

of discrete action-outcome pairs and to infer probabilistic outcomes of the next action. Felix Woolford proposed the Sensorimotor Sequence Reiterator [7]. We proposed the Enactive Cognitive Architecture [8]. These studies do not distinguish between generative and recognition models. They address the general question of how an enactive artificial agent can infer any *data structure* that it can reuse to generate adapted behavior through its lifetime interacting with the world.

In this paper, we delve further into exploring the connections between the mathematical formalism of active inference rooted in neuroscience tradition and the mathematical framework of artificial agent modeling derived from the computer science tradition, to lay the groundwork for envisioning a theory of *artificial enactive inference*. We do so by drawing relevant elements from the literature on Bayesian inference in Dynamic Bayesian Networks, and on autonomous-agent modeling from the literature on Partially Observable Markov Decision Process (POMDP). We adopt the term *data structure* from computer science terminology instead of *model* to avoid psychological connotations.

Sections 2 and 3 review the technical foundations of active inference and POMDPs. The reader familiar with these foundations may proceed directly to Section 4 that proposes the new SEMDP model, and Section 5 that demonstrates it in an early experimental robotics testbed. We believe that the foundational sections are nonetheless useful to introduce important definitions and to explain the need for the three-dimensional-world assumption introduced in Section 4.2. Section 2 develops and explains how Bayesian Networks, particularly Dynamic Bayesian Networks (DBNs), provide a formal basis for modeling enactive inference. They prepare the understanding of the concept of Markov Blanket used in Section 3.4. Section 3.2 shows that current methods for solving POMDPs require assumptions that are not satisfied in the context of enactive inference, which also led us to propose the three-dimensional-world assumption. Nevertheless, POMDPs offer a solid formal foundation for artificial enactive inference.

2. Inference in artificial agents

In this paper, we adopt the following notation. Bold uppercase letters, such as $\mathbf{A}$ and $\mathbf{B}$, are initially used to represent sets. Elements of these sets are denoted by lowercase letters, for example, $a \in \mathbf{A}$ and $b \in \mathbf{B}$. $\mathcal{P}(\mathbf{A})$

denotes the set of subsets of $\mathbf{A}$ (partition). $\Delta(\mathbf{A})$ denotes the *simplex* over $\mathbf{A}$, i.e., the set of probability distributions over $\mathbf{A}$.

Random variables are denoted by uppercase letters, for example, A and B. $P(A = a)$, or in short $P(A)$, denotes the probability that random variable A takes any particular value a. $P(B = b|A = a)$, or $P(B|A)$ denotes the conditional probability that B has the value b given that A has the value a. We use the assignment operator $:=$ to denote that the result of a function is assigned to a variable.

Notably, sets and variables play a similar role respectively in the contexts of set theory and probability theory. For example, when the environment transitions to a particular state, this is expressed in set theory by the fact that a particular state s is stochastically drawn from among all the possible states in set $\mathbf{S}$. In probability theory, this is expressed by the fact that variable S is assigned value s. Therefore, to simplify notation when the context permits and to imitate most of the POMDP and active inference literature [e.g., 9, 10], we merge these notations by generally omitting bold notation for sets beyond this short introduction.

2.1. Bayesian networks

Bayesian networks (BNs) [11] belong to the family of probabilistic graphical models. They combine principles from graph theory, probability theory, computer science, and statistics. More formally, BNs are defined as follows:

Definition 1 (Bayesian Network [12, 13]). *Consider U a finite set of random variables. A BN is a pair $B = \langle G, \Theta \rangle$ such that :*

- *$G = (V, E)$ is a directed acyclic graph with V the set of nodes representing variables from U, and E the set of edges where each edge represents the conditional dependencies among the corresponding variables.*
- *Θ is the set of conditional probability distributions, denoted $P(v|pa(v))$, of a variable $v \in V$ given its parents $pa(v)$.*

Figure 1 shows a simple BN representing that variable O depends on its parent variables S.

Figure 1: Bayesian network representing that "S influences O" or "O depends on its parent S" with conditional probabilities $P(O|S)$. The different probabilities involved with these two variables are linked through Bayes' theorem: $P(O)P(S|O) = P(S)P(O|S) = P(O, S)$.

2.2. Inference

The term *inference* refers to the process of reasoning or drawing conclusions about the uncertain variables in the network based on observed evidence or prior knowledge about some other variables. The main types of inference that interest us are *frequentist inference* and *probability inference*.

Definition 2 (Frequentist inference). *Frequentist inference is a method to estimate probability dependencies from data representing observation of events.*

If a list of events involving variables A and B is available, frequentist inference gives the estimated conditional probability $\hat{P}(B|A)$ of B given A through formula 1.

$$\hat{P}(B = b|A = a) = \frac{\text{Number of events in which } A = a \text{ and } B = b}{\text{Number of events in which } A = a} \tag{1}$$

For example, consider a robotic arm that can extend and detect impact with obstacles. We can estimate the probability of impact during extension by dividing the number of impacts during extension by the total number of arm extensions. Frequentist inference alone, however, does not allow an autonomous robot to infer the hidden cause of the impact: the presence of an obstacle at a certain location.

Definition 3 (Probability inference). *Probability inference is the process of computing the probability distribution of some variables in the network given the probability of other variables.*

Consider an agent observing its environment. Let us denote O the agent's observation and S the state of the environment. The fact that the state of the environment influences the observation can be represented by the BN in Figure 1. The statistics of S and O are linked by Bayes' theorem in Equation

2. If the other terms of the equation are known, probability inference allows inferring the state of the environment given the observation $P(S|O)$.

$$P(O)P(S|O) = P(S)P(O|S) = P(O,S) \tag{2}$$

For example, if our robotic arm only observes its environment through its "impact sensor", it could use probability inference to infer the state of the environment (the existence of obstacles at certain locations), in an analogous way as a blind person discovers their environment with a cane. This would require knowing the probability of observing an impact if there is an obstacle $P(O|S)$, the probability of each state $P(S)$, and the probability of each observation $P(O)$. If the cardinalities of S and O are large, this may also require unrealistic computational power because, in general, the computation of probability inference is NP-hard [14, 15].

In active inference theory, the joint probability distribution $P(O,S)$ over observations O and hidden causes S is called the *generative model*. The statistical mapping $P(S|O)$ from observable consequences O to hidden causes S is called the *recognition model* [1, p. 227]. To study how the agent can maintain these models over a succession of actions and observations, we use the formalism of Dynamic Bayesian Networks.

2.3. Dynamic Bayesian Networks

Dynamic Bayesian Networks (DBNs) [e.g., 16, 17] are an extension of Bayesian Networks used to represent the evolution of variables over discrete time. DBNs extend BNs to model probability distributions over semi-infinite series of random variables $Z_1, Z_2, ..., Z_t$.

In general, DBNs are first-order Markov models, meaning that the temporal dependency is specified over two time steps only as illustrated in Figure 2.

Definition 4 (Dynamic Bayesian Network). *A DBN is defined as a pair* $(B_0, B_\rightarrow)$*.* B_0 *is a BN which defines the prior* $P(Z_0)$*.* $B_\rightarrow$ *is a 2-step Temporal Bayesian Network (2TBN) that represents the temporal evolution of the DBN.*

A 2-step Temporal Bayesian Network (2TBN) is a BN divided into two *steps* that give the joint probability distribution $P(Z_t|Z_{t-1})$ of variables at step t given variables at step $t-1$.

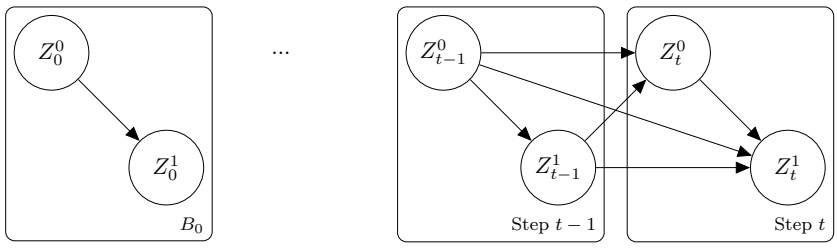

Figure 2: Example DBN with two variables Z^0 and Z^1. B_0 represents the *prior*. $B_{\rightarrow}$, made of *Step* $t-1$ and *Step* t, represents the two-step evolution of the DBN.

Some DBNs split Z_t into three distinct subsets: X_t, Y_t, and U_t, respectively, the input, hidden, and output variables. The probability distributions of variables in X_t can be inferred from observation using frequentist inference. The probability distributions of variables in U_t can be inferred from the DBN using probability inference.

When DBNs are used to represent the process of an agent interacting with its environment then some nodes are identified as *decision nodes*. Decision nodes are nodes whose values are set by the agent. Sometimes, the DBN is enriched with nodes that represent a scalar function such as a reward. Such enriched DBNs are called *influence diagrams*. In the rest of this article, the figures show influence diagrams in which the time subscripts associated with variables are omitted and the dependencies across time steps are shown with dotted arrows for clarity.

3. Interactions of artificial agents

In computer science and artificial intelligence, agents receive input data generated from the environment's state, compute and maintain internal data structures used to optimize their behavior and generate output data that trigger changes in the environment's state. The most broadly accepted formalism to describe this interaction cycle is a Partially Observable Markov Decision Process (POMDP), which we examine in this section.

3.1. POMDP Framework

A POMDP represents situations in which an artificial agent must make decisions to optimize some goal by interacting with an environment that it can only partially observe. At each interaction cycle t, the agent only receives a partial observation o_t generated from the state s_t hidden to the agent. The

designer defines a scalar reward function R to specify the agent's goal. In general cases, r_t is hidden to the agent. The following definition presents the POMDP formalism:

Definition 5 (POMDP). *A POMDP* Γ *is specified by a 6-tuple* $\Gamma = (S, D, O, p, q, R)$ *where:*

- S *is a finite set of states;*
- D *is a finite set of decisions sometimes called actions;*
- O *is a finite set of observations sometimes called signals;*
- $p : S \times D \to \Delta(S)$ *is a probabilistic transition function* $p(s_{t+1}|s_t, d_t)$ *that gives the probability distribution over the successor states* s_{t+1} *given a state* $s_t \in S$ *and a decision* $d_t \in D$*;*
- $q : S \times D \to \Delta(O)$ *is a probabilistic observation function* $q(o_{t+1}|s_t, d_t)$ *that gives the probability distribution over the observations* $o_{t+1} \in O$ *given a state* $s_t \in S$ *and a decision* $d_t \in D$ *;*
- $R : S \times D \to \mathbb{R}$ *is a step reward function.*

There exist different variants in the way observations are constructed [e.g., 18]. Many implementations directly map $q(o|s)$ from state to observation. In this case, the observation is *representational of the state* because it can be interpreted as representing a *feature* of the state. Some implementations use a mapping $q : S \to \mathcal{P}(O)$ from the state to the partition of O, each observation representing a different feature of the state. In the case of the mapping $q(o_{t+1}|s_t, d_t)$ used in Definition 5, the observation is *non-representational* of the state because the same state can produce different observations depending on the decision. In this case, o is more accurately named *signal* than *observation*. As enactive inference theory denies the notion that the brain has direct access to a representation of the environment, this paper abstains from assuming that o is representational and adopts the term signal.

The POMDP cycle is illustrated in Figure 3 and goes as follows:

- The POMDP is initialized with a probability distribution over the state space $p_0 \in \Delta(S)$ from which the initial state s_0 is drawn. The initial signal $o_0 \in O$ is drawn with probability $q(o_0|s_0)$ that does not involve a previous decision. This makes, in fact, o_0 a representational observation of s_0.

- At each step $t \geq 0$, the decider makes a decision $d_t \in D$. The next state s_{t+1} is drawn with probability $p(s_{t+1}|s_t, d_t)$. The next signal o_{t+1} is drawn with probability $q(o_{t+1}|d_t, s_{t+1})$. This decision determines the reward $r_t := R(s_t, d_t)$.

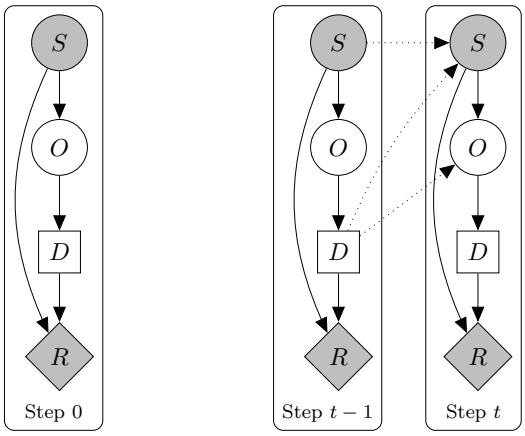

Figure 3: Influence diagram for a POMDP (adapted from [9], Figure 2) representing the state S, the signal O, the decision D, and the reward R. Gray nodes are hidden to the agent. In general cases, the agent only has indirect access to the reward R through the *objective criterion* not represented in the figure.

As the POMDP cycle unfolds, it generates a *history* of signals and decisions used to define the agent's *policy*.

Definition 6 (t-step history). *The t-step history is the sequence $h_t = (o_0, d_0, o_1, d_1, ..., d_{t-1}, o_t)$ of signals and decisions until step t. The set of all possible t-step histories is defined by: $H_0 = O$ and $H_t = H_{t-1} \times D \times O$.*

The total period over which the system is to be observed is called the *planning horizon* and is denoted by T. It can be a finite horizon $T = \{1, ..., |T|\}$ or an infinite horizon $T = \mathbb{N}$.

Definition 7 (policy). *The policy is a mapping $\sigma : \cup_{t \in T} H_t \to D$ that gives the decision d_t according to the history $h_t \in H_t$. The set of all policies is denoted Σ.*

POMDPs are widely used to model planning and control problems [19]. The designer defines the parameters (S, D, O, p, q, R) to fit a particular problem, and implements a *POMDP solution* to address it.

3.2. Solving POMDPs

In the POMDP literature, a *POMDP solution* is an algorithm that can compute an *optimal policy*. An optimal policy is a policy that optimizes an *objective criterion* defined by a *value function* [20]. Example value functions are the *long-run average reward* and the *discounted reward*. The former is the limit of the average reward. The latter is the sum of future discounted reward. It includes a discount factor expressing the agent's preference for immediate reward over longer-term reward.

Because of their computational complexity, it is only possible to compute such value functions in a few POMDPs. Indeed, finite-horizon POMDPs are PSPACE-complete [21], while infinite-horizon POMDPs are undecidable [20]. The majority of algorithms for exact planning in POMDPs work by optimizing the value function over all belief states. These algorithms can suffer from the *curse of dimensionality*, where the dimensionality of the planning problem is related to the number of states [22]. They can also suffer from the *curse of history*: the optimal action may depend on the entire history of actions and observations. Some recent algorithms, such as point-based algorithms, perform well [23] but the **computational complexity grows exponentially as a result of these two curses**.

While active inference can be used to optimize a value function, this is not its primary purpose. According to Ryan Smith et al.'s tutorial on active inference: "there are no additional variables labeled as 'reward' or 'values'. Instead, preferences are encoded within a specific type of prior probability distribution—which is often called a *prior preference distribution*" [10, p. 6]. The prior preference distribution $pr \in \Delta(O)$ associates a scalar preference with each signal. There is no value function that aggregates preferences over time. Instead, "the goal is to infer posterior beliefs over states and policies when conditioning on observations" [10, p. 11]. **In the context of active inference, solving a POMDP means designing an agent that learns to infer such beliefs**. For the agent, active inference is more a *task* to preform than a problem to solve.

3.3. Inferring belief states in POMDPs

The POMDP literature uses the term *belief state* to refer to the agent's estimation of the probability of being in any particular hidden state. Definition 8 provides a formal definition.

Definition 8 (t-step belief). *Given a policy $\sigma \in \Sigma$ and an initial belief $b_0 \in \Delta(S)$, the t-step belief b_t is defined by $b_t(s) = P_\sigma^{b_0}(S_t = s|H_t = h_t)$ which gives the agent's belief at step t that $S_t = s$ after some history $H_t = h_t$.*

Astrom [24] proposed a *belief update function* to compute the belief state b_t from b_{t-1}, starting from an initial belief b_0, in such a way that the belief state summarizes all the information gathered from the history. In this condition, the belief is a sufficient statistic for h_t [25], meaning that it can be used instead of the history with the advantage that it does not grow with time. The belief update function is defined as follows:

Definition 9 (Update function). *The belief update function is the function $\tau : O \times \Delta(S) \times D \to \Delta(S)$, which gives the successor belief $b_{t+1} := \tau(o_{t+1}, b_t, d_t)$.*

Let us examine τ to understand what prerequisite it entails. It is computed through Equation 3 that gives the belief to be in state s' on step $t+1$:

$$b_{t+1}(s') := \frac{1}{P(o_{t+1}|b_t, d_t)} q(o_{t+1}|s', d_t) \sum_{s \in S} p(s'|s, d_t) b_t(s) \tag{3}$$

with

$$P(o_{t+1}|b_t, d_t) := \sum_{s' \in S} q(o_{t+1}|s', d_t) \sum_{s \in S} p(s'|s, d_t) b_t(s)$$

Equation 3 uses the transition function p and the observation function q to compute the next belief state. When an engineer uses a POMDP to model a physical system, he or she knows p and q a priori, and uses them to implement τ. For research on autonomous agents, however, we do not wish to implement p and q because we study how the agent infers hidden states through experience without presupposed knowledge. Moreover, τ requires knowledge of the set of states S, which is generally large. Even when p and q are known, computing τ becomes intractable as the set of states grows [25].

The active inference literature has proposed a method to approximate the belief state through the minimization of *variational free energy* [26]. This method computes the posterior belief $b_{t+1}(s)$ after observation o_t through an iterative process of gradient descent of free energy. The variational free energy F is computed as a measure of similarity between the posterior belief and the generative model $P(O, S)$ according to equation 4, adapted from [10,

eq. 4]. The learning occurs through the iterative process of updating $b(s)$ on each interaction cycle.

$$F = \sum_{s \in S} b(s) ln \frac{b(s)}{P(O, S = s)} \tag{4}$$

Equation 4 shows that the variational free energy can only be computed if we presuppose the knowledge the generative model $P(O, S)$. While this presupposition is applicable in the case of well controlled experiments, it becomes unrealistic in the case of a robot in an open environment that is expected to learn without prior knowledge of its sensory apparatus. The sum and iterations over a large and unknown set of environmental states is also inapplicable.

3.4. Inference in unknown POMDPs

Very few studies addressed the inference problem in POMDPs when the set of states S, and transition and observation functions p and q are unknown to the agent. Alexander Mordvintsev [27] uses *differentiable optimization* to learn a Finite State Machine (FSM) representing an unknown environment whose input is the agent's actions and whose output is the agent's signals. It presupposes the maximum depth of temporal dependency, and the maximum number of states of the FSM. Our group [28] has proposed a method to learn a spatial representation of the environment. It presupposes that the enacted interactions can be localized in a two dimensional space and may cause the displacement of the agent's body.

Ramstead et al. refer to the concept of *Markov Blanket* to differentiate between the nodes that represent the agent's internal states and the nodes that represent the environment [1]. In a Bayesian Network, the Markov blanket of a set of nodes Z_0 is the set of nodes Z_M that isolates the nodes in Z_0 from all the other nodes in the network [29]. The Markov blanket principle states that the statistics of the nodes in Z_0 are entirely defined by the statistics of the nodes in Z_M, making the statistics of the nodes in Z_0 independent of the statistics of the other nodes of the graph as if they were hidden behind the Markov blanket. In the POMDP framework, the Markov blanket of the agent is made of the nodes that belong to the interface between the agent and the environment: the signal node O and the decision node D. **Because the nodes of the environment are hidden behind the agent's Markov blanket, a single node S holds the most complete statistics**

of the environment that the agent can possibly infer through its experience of interaction in the absence of predefined assumptions.

If the POMDP literature has not proposed tractable solutions for an artificial agent to perform inference in an unknown environment, and the Markov blanket argument coming from the BN literature suggests that inference is not possible beyond the single environment node S, then the question of artificial enactive inference remains unanswered. The following section puts forward a new perspective on this issue in light of this literature review.

4. Artificial enactive inference

Enactive Inference theory does not *a priori* posit representations of the agent's observational system and final goals. This contrasts with POMDP studies that posit these representations in the form of the tuple (S, D, O, p, q, R) that must be known *a priori* to infer belief states. In this section, we examine how to reconcile these two approaches to find the minimal mathematical assumptions that would be needed in computer science to account for the neuroscience theory of enactive inference.

From the study of POMDPs, we learned that the problem for an autonomous agent to infer the models introduced in Equation 2 is intractable in the general case. Moreover, the problem of maximizing a reward obtained from interaction is also intractable in the general case. The fact that natural organisms perform enactive inference with their limited computational resources suggests that enactive inference should be tractable. Therefore, implementing artificial agents capable of enactive inference is a completely different endeavor than implementing artificial agents that solve POMDPs.

This section proposes a new description of an agent interacting with its environment intended to formalize the artificial enactive inference task. To do so, we begin with presenting a simplification of the POMDP framework called the EMDP framework. This allows us to express assumptions that may make the artificial enactive inference task tractable, called the SEMDP framework.

4.1. Enactive Markov Decision Process (EMDP)

Compared to a POMDP, the main conceptual difference of an EMDP lies in the fact that the decider receives an *interaction* at the end of the cycle, rather than a *signal* at the beginning. We introduce the concept of *interaction* as a primitive of the model to represent a sensorimotor pattern

that involves both motor actions and sensory signals. As such, an interaction implements the fact that "perception and action cannot be pulled apart" stressed by the theory of enactive inference introduced in Section 1. In constructivist learning theory, an interaction corresponds to a Piagetian sensorimotor *scheme*. In a robot, an interaction can be implemented through a *control loop* involving actuator motion and sensory feedback [30]. Because interactions depend both on the decision and on the state, they can be seen as the counterpart of the couple (decision, signal) in a POMDP. Section 5 will provide an example.

Another important difference is that EMDPs do not contain a reward function. As introduced in Section 3.2, the purpose of enactive inference is not to maximize reward. The EMDP formalism makes no commitment on the behavior selection mechanism that drives the agent. For example, Jacqueline Gottlieb [e.g. 31] has recently proposed a review of various information seeking motivational principles which we believe could be compatible with the EMDP formalism. It is nonetheless possible to define a preference distribution $pr \in \Delta(I)$ that attributes a scalar *preference* to each interaction similar to that introduced in Section 3.2. This defines a kind of self-motivation that we have called *interactional motivation* in other experiments [32].

As the sets of actions and interactions are finite, we can define EMDPs on very general state spaces. More specifically, let us proceed by defining the characteristics of an EMDP:

Definition 10 (EMDP). *An Enactive Markov Decision Process is specified by a 5-tuple* $\Gamma_E = (S, D, I, p, q)$ *where:*

- S *is a countable or uncountable set of states;*
- D *is a finite set of decisions;*
- I *is a finite set of interactions;*
- $q : S \times D \to \Delta(I)$ *is a probabilistic transition function* $p(i_t|s_t, d_t)$ *that gives the probability distribution over the interactions* $i_t \in I$*, given a state* $s_t \in S$ *and a decision* $d_t \in D$*;*
- $p : S \times I \to \Delta(S)$ *is a probabilistic transition function* $p(s_{t+1}|s_t, i_t)$ *that gives the probability distribution over the successor state* $s_{t+1} \in S$*, given a state* $s_t \in S$ *and an interaction* $i_t \in I$*;*

The EMDP cycle goes as follows:

- The EMPD is initialized with an initial state $s_0 \in S$.
- At each step $t \geq 0$, The agent makes some decision $d_t \in D$. The enacted interaction i_t is drawn with probability $q(i_t|s_t, d_t)$. The next state s_{t+1} is drawn with probability $p(s_{t+1}|s_t, i_t)$.

Like a POMDP, an EMDP rests upon the assumption that the same interaction enacted in the same state always yields the succession state with the same probability. The EMDP can, therefore, be represented as a 2-step DBN illustrated in Figure 4.

In an EMDP, the enacted interaction i_t contains information on the previous decision d_t. The notion of history introduced in section 3.1, is therefore simplified in the form of the sequence of enacted interactions $h_t = \{i_0, ..., i_{t-1}\}$ before step t. As with POMDPs, the history contains the relevant information for the decision-making process. It constitutes the Markov blanket that hides the environment's structure to the agent.

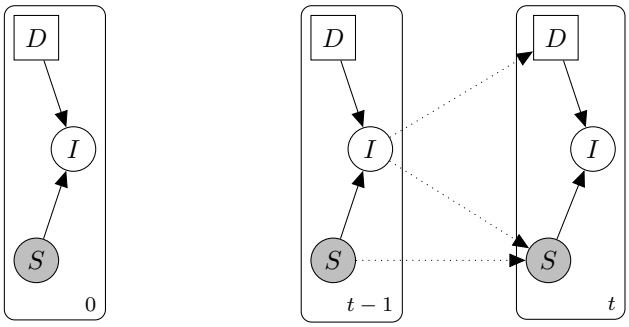

Figure 4: Influence diagram for an Enactive Markov Decision Process (EMDP). The interaction I_t depends on the decision D_t and the state S_t. The previous enacted interaction I_{t-1} influences the decision D_t and the state S_t. The previous state S_{t-1} influences the current state S_t. The nodes I constitute the Markov blanket that hides the nodes S (in gray) to the agent.

Another notable difference between EMDPs and POMDPs pertains to the belief and the methodology of information collection from the environment. In EMDPs, both the transition and observation functions are unknown. Therefore, it is unfeasible to update the belief. Instead, within the EMDPs framework, the agent gathers information from its environment through *enacted interactions*.

4.2. Spatial Enactive Markov Decision Process (SEMDP)

Living organisms have evolved in a 3D world where understanding the location of the organism within that 3D world and being able to localize objects in that 3D world has been essential for the survival of the species. It is not surprising, therefore, that mechanisms have evolved to address the representation of self and objects in 3D space [33, 34, 35, 36, 37, 38]. Equally, much effort has gone into producing biomimetic counterparts for robots [39, 40, 41, 42, 43, 44]. To account for how such organisms perform enactive inference in the physical world, we propose the Spatial Enactive Markov Decision Process (SEMDP) formalism. The SEMDP is a specialization of the EMDP that encodes the assumption that the set of states S is a *three-dimensional world*. A 3D world is a 3D Euclidean space in which objects are represented as sets of points upon which geometrical affine transformations can be performed. A 3D Euclidean space is a three-dimensional real vector space $\mathcal{E} = \mathbb{R}^3$ associated with Euclidean distance.

We define a SEMDP by extending the EMDP model as a tuple $\Gamma_S = (S, D, I, \mathcal{E}, \mathcal{I}, p, q)$ where $\mathcal{E}$ is a 3D Euclidean space, and $\mathcal{I}$ is the set of direct isometries over $\mathcal{E}$. A direct isometry, or a *rigid motion*, is an affine transformation that preserves distances and angles, i.e., a translation, a rotations, or a combination of both. As with the EMDP cycle introduced in Section 4.1, the SEMDP cycle is shown in Figure 5 and goes as follows:

- The SEMDP is initialized with an initial state $s_0 \in S$.
- At each step $t \geq 0$, The agent makes some decision $d_t \in D$. The enacted interaction $i_t \in I$, its position $\epsilon_t \in \mathcal{E}$, and the isometry $\iota_t \in \mathcal{I}$ are drawn with probability $q(i_t, \epsilon_t, \iota_t | s_t, d_t)$. The next state s_{t+1} is drawn with probability $p(s_{t+1} | s_t, i_t, \epsilon_t, \iota_t)$.

The agent implements the assumption that it controls a mobile robot in the 3D world. The point ϵ, and the isometry ι respectively indicate the position of the enacted interaction i, and the displacement of the robot during the enaction of the interaction. Both ϵ and ι are encoded in egocentric coordinates relative the robot's body. For example, if the robot impacts an obstacle while enacting an interaction, the `impact` interaction is localized at the position of the robot's bumper, and the isometry represents the translation of the robot before being blocked by the obstacle. This 3D-world assumption finds support in the observation that most animals can localize

their interactions in their surrounding environment and can detect their own displacement (e.g., through vestibular signal and optic flow). Jeff Hawkins and his group express this hypothesis as follows:

> "Everything is perceived at a location. As we attend to each object, we perceive the distance and direction from ourselves to these objects and where they are relative to each other. The sense of location and distance is inherent to perception, it occurs without effort or delay" [45, p. 12].

Note that the agent does not have access to any putative "absolute position" of the robot in the environment.

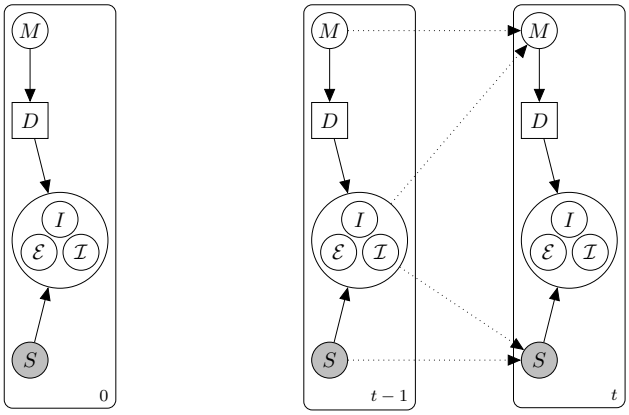

Figure 5: Representation of a Spatial Enactive Markov Decision Process (SEMDP). The memory state M influences the decision D. The interaction I is associated with its relative position $\mathcal{E}$ and with the displacement $\mathcal{I}$ of the robot. The state S (gray nodes) representing the environment is hidden to the agent.

As in a POMDP and an EMDP, the history constitutes the relevant information for the decision-making process. In a SEMDP, it becomes defined as the sequence of enacted interactions with their spatial position $\epsilon \in \mathcal{E}$ and the robot's displacement $\iota \in \mathcal{I}$ before step t: $h_t = \{i_0, \epsilon_0, \iota_0, ..., i_{t-1}, \epsilon_{t-1}, \iota_{t-1}\}$. The history is stored in memory M organized spatially.

The memory update function becomes defined as $\tau : M \times I \times \mathcal{E} \times \mathcal{I} \to M$ and gives the successor memory state $m_{t+1} := \tau(m_t, i_t, \epsilon_t, \iota_t)$. At the end of step t, τ applies the inverse isometry ι_t^{-1} to the position of all previously memorized enacted interactions: $\epsilon_n := \iota_t^{-1}(\epsilon_n)$ for $n \in \{0, ..., t-1\}$. The reason τ uses the inverse isometry is that the displacement of the interaction

relative to the robot is the inverse of the displacement of the robot relative to the environment. And then τ adds the new enacted interaction i_t at position ϵ_t to memory, as will be shown in the cognitive architecture presented next.

4.3. Cognitive architecture for enactive inference in a SEMDP

We designed a brain-inspired cognitive architecture to implement the policy of a SEMDP capable of enactive inference. Its code is shared online [46]. We drew inspiration from areas used for spatial representation in the mammalian brain [47], specifically the superior colliculus associated with egocentric memory and hexagonal grid cells in the hippocampus associated with allocentric memory.

Figure 6 outlines the modules of the architecture. At the end of step t, the enacted interaction i_t is added to egocentric spatial memory using its position ϵ_t. The positions of the previous interactions are updated using ι_t^{-1} as introduced above. Simultaneously, i_t is also added to hierarchical sequential memory. The sequence learning mechanism is a kind of sensorimotor sequence reiterator as some authors have proposed [7] and as we have described in previous articles [8, 48]. In essence, it records a series of enacted interactions and organizes them in a hierarchy in which longer sequences are made of series of shorter sequences. When a sub-sequence is recognized in the short-term history, its following sub-sequence is proposed for future enaction. This mechanism results in the agent learning habits of interactions. It can also be *seeded* with "innate" behaviors.

Egocentric spatial memory is converted into allocentric spatial memory by reference to a fixed point in space. By default, the initial point where i_0 was enacted is used as the reference. The architecture examines the hypothesis that closely located interactions may be *afforded* by a single physical object present in the environment. For this examination, the architecture creates tuples that associate interactions with their allocentric position. We call these tuples *affordances*. Next, the architecture gathers affordances located next to each other in sets that may describe the physical object "as the agent experiences it through interaction". This learning process that goes from interactions to objects may seem counter-intuitive because many studies proceed in the opposite direction by learning affordances of presupposed objects. It is, however, the spirit of active inference to infer the causes in the environment (hypothetical objects) that explain the agent's sensorimotor experience. We use the term *phenomenon* to refer to a set of affordances intended to represent a physical object. This usage of the term complies with

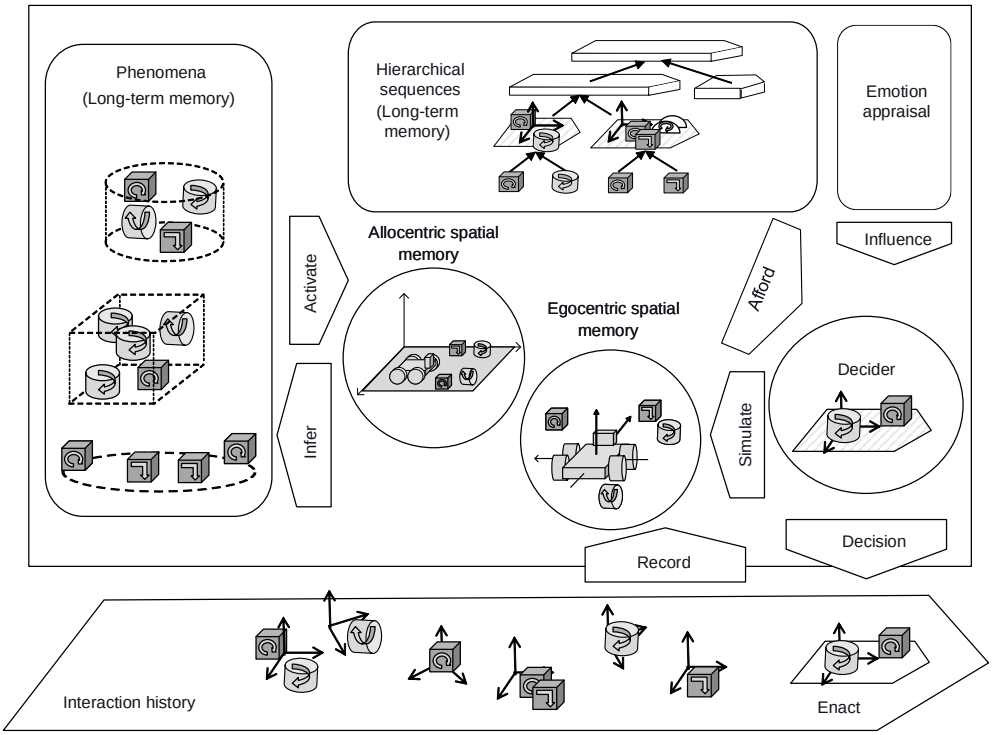

Figure 6: Cognitive architecture for SEMDP. Bottom: the history h_n: for $n \in \{0, t\}$. Enacted interactions i_n associated with their position ϵ_n and isometry ι_n are represented as small 3D blocks with arrows. The cognitive architecture incorporates the long-term memory of hierarchical sequences of interactions (top), egocentric and allocentric working spatial memories (center), and the long-term memory of inferred phenomena (left). Right: the Decider makes decision d_t on the basis of the states of all the memory modules. Top right: the emotion appraisal module is not used in this study except for simulating emotional states displayed by the robot's color led.

its common-sense usage: "something" that a cognitive being perceives in the environment. Thórisson offers a more technical definition of the term phenomenon that also matches this usage: "any useful grouping of a subset of spatio-temporal patterns experienced by an agent in an environment" [49, p. 8]. The phenomenon sets are the product of the enactive inference process.

This cognitive architecture implements a *finite memory policy*: the number of enacted interactions stored in memory M grows linearly with time. It can be bounded by forgetting interactions older than an arbitrary step limit l since its purpose is not to optimize a value. Enacted interactions are duplicated in the various modules of the architectures, but the maximum size remains bounded by the number of modules multiplied by l. Therefore, the cognitive architecture complies with Definition 11 of a finite-memory policy proposed, for example, by Chatterjee et al. [50]. In practice, the required memory size remains orders of magnitude lower than the memory capacity of a regular personal computer.

Definition 11 (Finite-Memory policy). *A policy σ is said to have finite-memory if it can be modeled by a finite-state transducer. Formally, $\sigma = (\sigma_u, \sigma_a, M, m_0)$, where M is a finite set of memory states, m_0 is the initial memory state, $\sigma_a : M \to D$ is the decision selection function, and $\sigma_u : M \times D \times O \to M$ is the memory update function.*

5. Experiment

We built a mobile robot called *PetitCat* based on the "Robot car" of brand Osoyoo [51]. It includes omnidirectional wheels allowing axial and lateral translations, a pivoting head holding an ultrasonic telemeter, a bar of luminosity sensors directed to the floor in front of the robot, and a WiFi module. We added an inertial measurement unit, a color sensor directed to the floor underneath the robot, and an RGB LED on top. Figure 7 illustrates this configuration.

5.1. Settings

The PC implements the cognitive architecture that selects decision d_t and sends it to the robot via WiFi in the *decision packet*. The decision packet also includes additional information not detailed in this paper such as instructions to move the head, and color of the RGB LED. Table 1 lists the set D of possible decisions. The Arduino board decodes the decision and

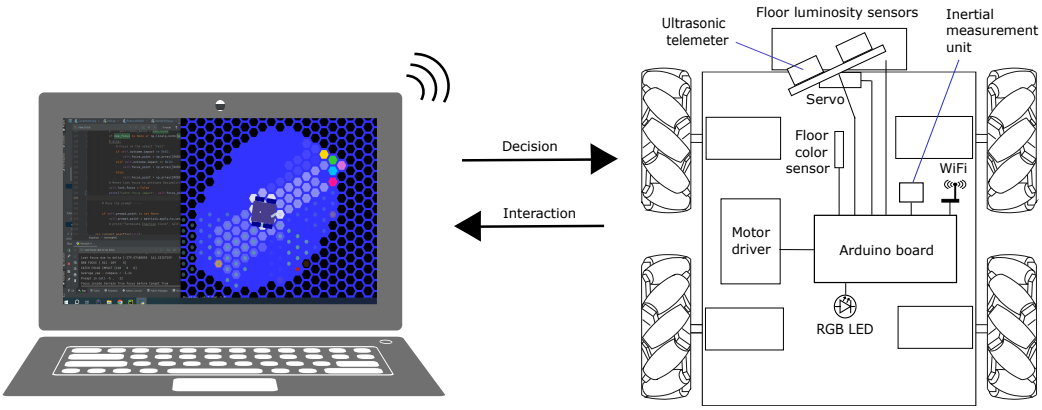

Figure 7: Hardware configuration showing the PetitCat robot controlled by an Arduino board. Actuators are the motors controlling the omnidirectional wheels, the servo controlling the head direction, and the RGB LED displaying a representation of the robot's internal state. Sensors are the ultrasonic telemeter, the floor luminosity sensor bar, the inertial measurement unit, and the floor color sensor. The PC implements the cognitive architecture and remote controls the robot through WiFi by sending *decision packets*, and receiving *interaction packets* back from the robot.

implements the control loop that enacts the corresponding interaction over a predefined default time span or angle span.

Table 1: Set of available decisions D

Decision	**Description**	Default span
`turn_left`	Rotate in the spot to the left	$angle = \pi/4$
`backward`	Translate backward	$duration = 1\ s$
`turn_right`	Rotate in the spot to the right	$angle = -\pi/4$
`swipe_left`	Translate to the left	$duration = 1\ s$
`swipe_right`	Translate to the right	$duration = 1\ s$
`forward`	Translate forward	$duration = 1\ s$

To enact the interaction, the Arduino board runs a timer and a control loop that continuously checks the floor luminosity, the yaw, the linear acceleration, and the ultrasonic echo distance. This loop also constantly adjusts the direction of the head in search of the shortest echo distance. When an interruption event is triggered, the robot stops the interaction and returns the *interaction packet* to the PC, containing information about the enacted interaction and the event. Table 2 summarizes the possible events. The `floor`

event additionally triggers a slight automatic withdrawal that prevents the robot from crossing the black line.

Table 2: Set of interrupting events

Code	Description
`floor`	Drop in floor luminosity indicating a black line on the floor
`impact`	Strong deceleration indicating an impact against an obstacle
`blocked`	Weak acceleration on startup indicating that the robot is blocked by an obstacle
`left_echo`	Echo distance below 150 mm and head angle above 30° indicating an obstacle to the left
`front_echo`	Echo distance below 150 mm and head angle between -30° and 30° indicating an obstacle in front
`right_echo`	Echo distance below 150 mm and head angle below -30° indicating an obstacle to the right
`span_up`	The target span has been reached (rotation angle or duration)

Upon reception, the PC transcribes the interaction packet into the enacted interaction $i_t \in I$, its position $\epsilon_t \in \mathcal{E}$, and the robot's displacement $\iota_t \in \mathcal{I}$. The set of enacted interactions is made of the combination of the possible decisions and their interrupting events, making 6 decisions x 7 events = 42 interactions. The position ϵ_t depends on the type of the event. Interactions involving the `floor`, `impact`, and `blocked` events are localized at the front edge of the robot. Interactions caused by echoes are localized at the distance of the echo, in the direction of the head. The isometry ι_t representing the robot's displacement is a combination of the yaw and translation on the floor. The yaw is measured by the inertial measurement unit with the relatively good precision of ±2%. The translation is obtained by multiplying the robot's speed by the duration of the interaction before interruption (approximately 200 mm/s). It has a poor precision of ±20%. While the cognitive architecture can handle positions and displacements in the three-dimensional space, this experiment only uses the two-dimensional horizontal plane.

We seeded the sensorimotor sequence reiterator with two basic behaviors. Behavior 1 consists of turning to the left when a `right_echo` event is triggered. Behavior 2 consists of moving forward as long as no `floor`, `impact`, `left_echo`, or `front_echo` event is triggered, and swiping to the left after those events are triggered. These simple behaviors are sufficient to infer the shape of simple objects and the arena as reported next.

5.2. Results

With the seed behaviors presented above, when the robot is switched on, it repeatedly decides to move forward until a `floor` or `echo` event is triggered. After that, the robot swipes to the left and turns. This behavior

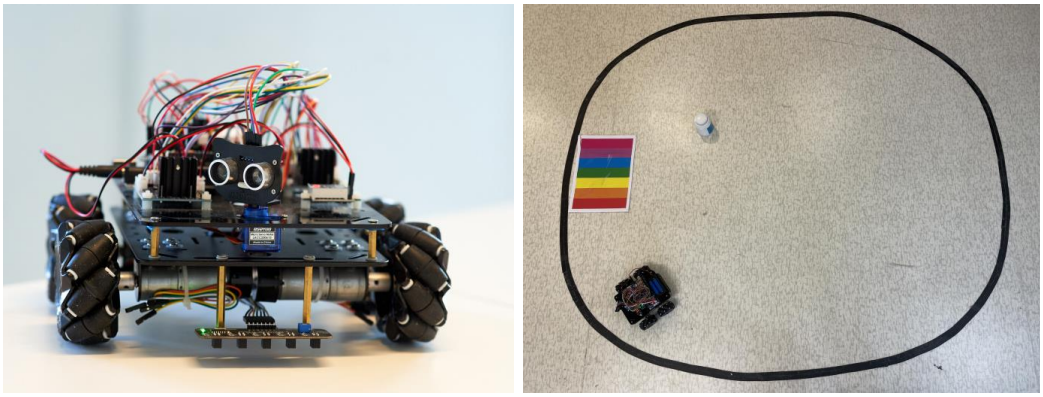

Figure 8: Left: The PetitCat robot. Right: The robot and a bottle in the arena delimited by a black line.

results in the robot circling around the "phenomena" that it encounters while constructing data structures in spatial memory that account for the shape of these phenomena. For example, when the phenomenon is a bottle placed on the floor, the robot constructs the data structures shown in Figure 9 after less than a hundred interactions. The shapes inferred by the robot, while imprecise, allow discriminating between different phenomena. Detailed descriptions are available in a previous article [52].

When the phenomenon is the arena delimited by the black line, the robot circles inside it. An example run can be seen in video [53]. Figure 10 (left) shows a graphical representation of the arena constructed in phenomenon memory at step $t = 58$ when the robot completed its tour. Each bold segment represents an affordance based upon an interaction with the black line $i_n, n \in \{0, ..., t\}$ at position ϵ_n moved by the inverse isometries $\iota_m^{-1}, m \in \{n+1, ..., t\}$ accumulated over steps. The thin oval black line shows the interpolated border of the arena. Over time, the imprecision in the robot's displacements accumulates, and the position of interactions in memory drifts. The robot uses the color patch on the floor detected through the color sensor as an absolute reference to correct the drift when the robot returns to the color patch. The robot also infers that the space outside the border is out of reach, which is represented by black hexagons in Figure 10 (right).

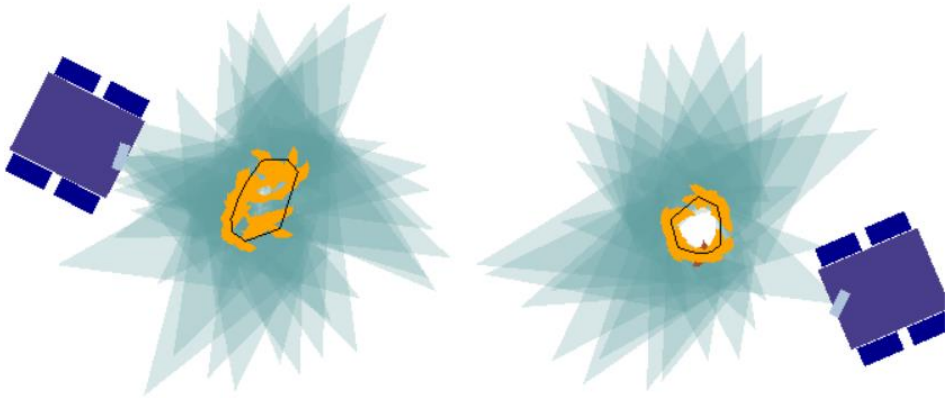

Figure 9: Graphical representations of phenomena encoded as sets of affordances. Each affordance is represented by a blue-gray triangle (cone of echo-localization) and an orange half-circle (estimated position of echo). Black line: the object's outline inferred by the robot. Right: Phenomenon constructed from a bottle of diameter 100 mm. Left: Phenomenon constructed from two bottles next to each other. The robot "sees" them as a single solid object.

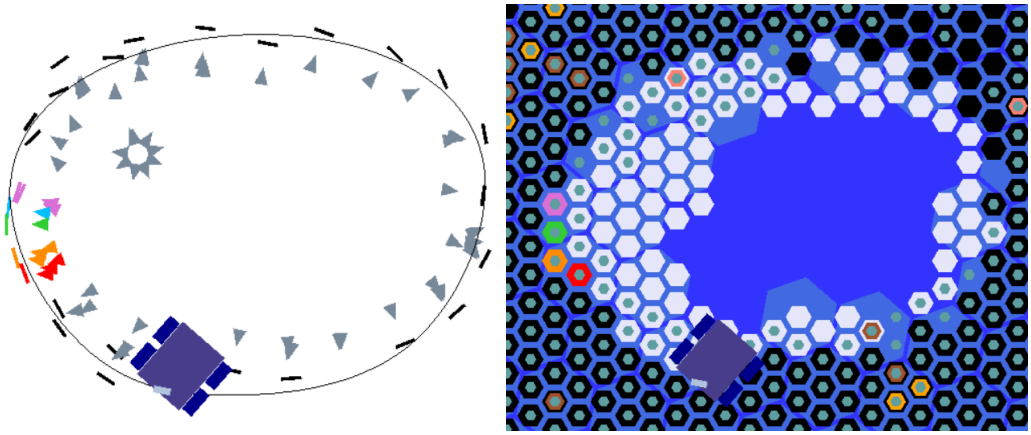

Figure 10: Left: graphical representation of the arena in phenomenon memory at step 58, flipped to the same direction as in Figure 8 for easier understanding. Bold black or colored segments: affordances constructed from interactions with the black line detected by luminosity and color sensors. Gray or colored triangles: position of the robot on each interaction cycle. Thin black line: the shape of the arena inferred by the robot. Right: alocentric memory represented as an hexagonal grid. Black hexagons: the area that the robot has inferred is inaccessible.

6. Conclusion

Our experimental results show that a robot can perform artificial enactive inference as defined within the framework of Spatial Enactive Markov Decision Process. The robot exploits the presupposition that it can localize its interactions in its surrounding space, and sense its own displacements. This assumption is built into the robot through the choice of sensors, and hard-coded in its cognitive architecture. What is not hard-coded and left for the robot to infer is ontological knowledge about the objects that populate the environment.

This study merely begins to explore the intricacies involved in inferring complex knowledge through interactive experiences. Perhaps the most intriguing question is that of implementing motivational drives in autonomous robots. Friston has proposed the free energy principle according to which the agent makes the decisions that minimize *prediction error*. Prediction error is the difference between sensory signal anticipated through the generative model and sensory signal actually measured.

Our cognitive architecture computes prediction errors of each sensory

signal. For example, when the robot decides to move forward for one second toward an object, it anticipates its displacement based on its estimated speed, and tries to predict the distance to the object that will be measured after the displacement. The echo-localization prediction error is the difference of this anticipation computed at the beginning of the interaction cycle minus the signal measured at the end of the interaction cycle. It may depend on various factors such as irregularities on the floor, imprecision of the sensor itself, battery level which impacts the robot's speed, and of course, whether the object has moved since the last measure. We cannot find a unique variable whose optimization would correlate with minimization of prediction error of all the sensors.

To paraphrase Josha Bach's vision on how motivational drives may work in biological-organisms [54, t.c. 1:18:00], we imagine that a newly-switched-on robot should be driven by numerous intrinsic drives that may involve individual and social preferences as well as cognitive needs. These drives would be seeded in the robot's initial cognitive architecture as reflex that direct the robot until it can form higher goals. We don't expect that this process could be achieved only by optimizing a unique scalar value. Instead, it would require intelligence. The robot would have to make experiments and draw inferences starting with hard-coded first-principle reasoning mechanisms. Such mechanisms do not require an initial personal self, but they may be a prerequisite for the construction of the self.

Some authors like György Buzsáki have claimed that the human brain recycles spatial-computation capacities inherited through phylogenetic evolution to perform reasoning upon abstract concepts [55]. This suggests that future progress in designing SEMDP agents may lead to autonomous robots capable of constructing abstract knowledge grounded on their own sensorimotor experience.

Acknowledgment

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0035. It benefited from discussions with Howard Schneider and Ricardo Gudwin, and from the work of Pierre Martin and Marouane Toumi.

References

[1] M. J. Ramstead, M. D. Kirchhoff, K. J. Friston, A tale of two densities: active inference is enactive inference, Adaptive Behavior 28 (4) (2020).

[2] K. Friston, R. J. Moran, Y. Nagai, T. Taniguchi, H. Gomi, J. Tenenbaum, World model learning and inference, Neural Networks 144 (2021) 573–590. doi:10.1016/j.neunet.2021.09.011.

[3] D. D. Hutto, E. Myin, Radicalizing Enactivism: Basic Minds Without Content, Cambridge, MA: MIT Press, 2013.

[4] F. J. Varela, E. Thompson, E. Rosch, The embodied mind: cognitive science and human experience, 14th Edition, MIT Press, 1993.

[5] J. K. O'Regan, A. Noë, A sensorimotor account of vision and visual consciousness, Behavioral and Brain Sciences 24 (5) (2001) 939–973. doi:10.1017/S0140525X01000115.

[6] A. Maye, A. K. Engel, Extending sensorimotor contingency theory: prediction, planning, and action generation, Adaptive Behavior 21 (6) (2013) 423–436. doi:10.1177/1059712313497975.

[7] F. M. G. Woolford, M. D. Egbert, A precarious sensorimotor sequence reiterator for modelling enactive habits, in: ALIFE 2020: The 2020 Conference on Artificial Life, MIT Press, 2020, pp. 771–779. doi:10.1162/isal_a_00249.

[8] O. L. Georgeon, J. B. Marshall, R. Manzotti, ECA: An enactivist cognitive architecture based on sensorimotor modeling, Biologically Inspired Cognitive Architectures 6 (2013) 46–57. doi:10.1016/j.bica.2013.05.006.

[9] E. A. Hansen, An integrated approach to solving influence diagrams and finite-horizon partially observable decision processes, Artificial Intelligence 294 (2021) 103431. doi:10.1016/j.artint.2020.103431.

[10] R. Smith, K. J. Friston, C. J. Whyte, A step-by-step tutorial on active inference and its application to empirical data, Journal of Mathematical Psychology 107 (2022) 102632. doi:10.1016/j.jmp.2021.102632.

[11] I. Ben-Gal, Bayesian networks, Encyclopedia of statistics in quality and reliability 1 (2008).

[12] N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifiers, Machine learning 29 (1997) 131–163.

[13] K. Murphy, A brief introduction to graphical models and bayesian networks, 1998, Available electronically at http://www. cs. ubc. ca/ murphyk/Bayes/bnintro. html (1998).

[14] G. F. Cooper, The computational complexity of probabilistic inference using bayesian belief networks, Artificial intelligence 42 (2-3) (1990) 393–405.

[15] P. Dagum, M. Luby, Approximating probabilistic inference in bayesian belief networks is np-hard, Artificial intelligence 60 (1) (1993) 141–153.

[16] T. Dean, K. Kanazawa, A model for reasoning about persistence and causation, Computational intelligence 5 (2) (1989) 142–150.

[17] K. Murphy, Dynamic bayesian networks: Representation, inference and learning, Phd thesis, University of California, Berkeley (2002).

[18] K. Chatterjee, M. Chmelik, M. Tracol, What is decidable about partially observable markov decision processes with ω-regular objectives, Journal of Computer and System Sciences 82 (5) (2016) 878–911.

[19] A. R. Cassandra, A survey of pomdp applications, in: Working notes of AAAI 1998 fall symposium on planning with partially observable Markov decision processes, Vol. 1724, 1998.

[20] O. Madani, S. Hanks, A. Condon, On the undecidability of probabilistic planning and infinite-horizon partially observable markov decision problems, in: AAAI/IAAI, 1999, pp. 541–548.

[21] C. H. Papadimitriou, J. N. Tsitsiklis, The complexity of markov decision processes, Mathematics of operations research 12 (3) (1987) 441–450.

[22] L. P. Kaelbling, M. L. Littman, A. R. Cassandra, Planning and acting in partially observable stochastic domains, Artificial intelligence 101 (1-2) (1998) 99–134.

[23] W. Lee, N. Rong, D. Hsu, What makes some pomdp problems easy to approximate?, Advances in neural information processing systems 20 (2007).

[24] K. J. Åström, Optimal control of markov processes with incomplete state information, Journal of mathematical analysis and applications 10 (1) (1965) 174–205.

[25] R. D. Smallwood, E. J. Sondik, The optimal control of partially observable markov processes over a finite horizon, Operations research 21 (5) (1973) 1071–1088.

[26] K. Friston, The free-energy principle: a unified brain theory?, Nature Reviews Neuroscience 11 (2) (2010) 127–138. doi:10.1038/nrn2787.

[27] A. Mordvintsev, Differentiable finite state machines (2022).
URL https://google-research.github.io/self-organising-systems/2022/diff-fsm/

[28] S. L. Gay, A. Mille, O. L. Georgeon, A. Dutech, Autonomous construction and exploitation of a spatial memory by a self-motivated agent, Cognitive Systems Research 41 (2017) 1–35. doi:10.1016/j.cogsys.2016.07.004.

[29] J. Pearl, Probabilistic reasoning in intelligent systems : networks of plausible inference, San Mateo, Calif. : Morgan Kaufmann Publishers, 1988.

[30] O. L. Georgeon, A. Riegler, CASH only: Constitutive autonomy through motorsensory self-programming, Cognitive Systems Research 58 (2019) 366–374. doi:10.1016/j.cogsys.2019.08.006.

[31] J. Gottlieb, P.-Y. Oudeyer, Towards a neuroscience of active sampling and curiosity, Nature Reviews. Neuroscience 19 (12) (2018) 758–770. doi:10.1038/s41583-018-0078-0.

[32] O. L. Georgeon, J. B. Marshall, S. Gay, Interactional motivation in artificial systems: Between extrinsic and intrinsic motivation, in: IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL), IEEE, 2012, pp. 1–2. doi:10.1109/DevLrn.2012.6400833.

[33] S. Poulter, T. Hartley, C. Lever, The neurobiology of mammalian navigation, Current Biology 28 (17) (2018) R1023–R1042, publisher: Elsevier.

[34] W. Tang, J. D. Shin, S. P. Jadhav, Multiple time-scales of decision-making in the hippocampus and prefrontal cortex, Elife 10 (2021) e66227, publisher: eLife Sciences Publications, Ltd.

[35] S. Garg, T. Fischer, M. Milford, Where is your place, visual place recognition?, arXiv preprint arXiv:2103.06443 (2021).

[36] F. Savelli, J. J. Knierim, Origin and role of path integration in the cognitive representations of the hippocampus: computational insights into open questions, Journal of Experimental Biology 222 (Suppl_1) (2019) jeb188912, publisher: The Company of Biologists Ltd.

[37] K. Balakrishnan, R. Bhatt, V. Honavar, A computational model of rodent spatial learning and some behavioral experiments, in: Proceedings of the Twentieth Annual Meeting of the Cognitive Science Society, 2022, pp. 102–107.

[38] N. Yu, Y. Zhai, Y. Yuan, Z. Wang, A bionic robot navigation algorithm based on cognitive mechanism of hippocampus, IEEE Transactions on Automation Science and Engineering 16 (4) (2019) 1640–1652, publisher: IEEE.

[39] Z. Gao, Q. Shi, T. Fukuda, C. Li, Q. Huang, An overview of biomimetic robots with animal behaviors, Neurocomputing 332 (2019) 339–350, publisher: Elsevier.

[40] W. X. Schneider, J. Albert, H. Ritter, Enabling cognitive behavior of humans, animals, and machines: A situation model framework, ZiF-Mitteilungen 1 (2020) 21–34.

[41] F. Yu, J. Shang, Y. Hu, M. Milford, NeuroSLAM: a brain-inspired SLAM system for 3D environments, Biological cybernetics 113 (2019) 515–545, publisher: Springer.

[42] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, S. Levine, Learning agile robotic locomotion skills by imitating animals, arXiv preprint arXiv:2004.00784 (2020).

[43] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, L. Carlone, Kimera: From SLAM to spatial perception with

3D dynamic scene graphs, The International Journal of Robotics Research 40 (12-14) (2021) 1510–1546, publisher: SAGE Publications Sage UK: London, England.

[44] T. E. Behrens, T. H. Muller, J. C. Whittington, S. Mark, A. B. Baram, K. L. Stachenfeld, Z. Kurth-Nelson, What is a cognitive map? Organizing knowledge for flexible behavior, Neuron 100 (2) (2018) 490–509, publisher: Elsevier.

[45] J. Hawkins, M. Lewis, M. Klukas, S. Purdy, S. Ahmad, A framework for intelligence and cortical function based on grid cells in the neocortex, Frontiers in Neural Circuits 12 (2019).

[46] O. L. Georgeon, Petitcat project repository (2024).
URL `https://github.com/UCLy/INIT2/`

[47] R. M. Grieves, K. J. Jeffery, The representation of space in the brain, Behavioural Processes 135 (2017) 113–131. doi:https://doi.org/10.1016/j.beproc.2016.12.012.

[48] P. Robertson, R. Laddaga, A biologically inspired spatial computer that learns to see and act, in: Spatial Computing Workshop, SASO 2009, San Francisco, 2009.

[49] K. R. Thórisson, The 'Explanation Hypothesis' in general self-supervised Learning, International Workshop in Self-Supervised Learning, 2021.

[50] K. Chatterjee, R. Saona, B. Ziliotto, Finite-memory strategies in pomdps with long-run average objectives, Mathematics of Operations Research 47 (1) (2022) 100–119.

[51] Osoyoo, M2.0 metal chassis mecanum wheel robotic (2022).
URL `https://osoyoo.com/2022/07/05/v2-metal-chassis-mecanum-wheel-robotic-for-`

[52] O. L. Georgeon, J. R. Vidal, T. Knockaert, P. Robertson, Simultaneous localization and active phenomenon inference (SLAPI), in: K. R. Thórisson (Ed.), Proceedings of the Third International Workshop on Self-Supervised Learning, Vol. 192 of Proceedings of Machine Learning Research, PMLR, 2022, pp. 77–88.

[53] O. L. Georgeon, Robot exploring the arena (2023).
URL https://youtu.be/rKYiXNGiyiE

[54] L. Fridman, Joscha bach: Life, intelligence, consciousness, AI & the future of humans (2023).
URL https://youtu.be/e8qJsk1j2zE

[55] G. Buzsáki, E. I. Moser, Memory, navigation and theta rhythm in the hippocampal-entorhinal system, Nature Neuroscience 16 (2) (2013) 130–138. doi:10.1038/nn.3304.