Simultaneous Localization and Active Phenomenon Inference (SLAPI)

Olivier L. Georgeon[1] OGEORGEON@UNIV-CATHOLYON.FR
Juan R. Vidal[2] JVIDAL@UNIV-CATHOLYON.FR
[1,2] *UR CONFLUENCE: Sciences et Humanités (EA 1598), UCLy, Lyon Catholic University, France*
Titouan Knockaert TITOUAN.KNOCKAERT@GMAIL.COM
Université Claude Bernard Lyon 1, LIRIS CNRS UMR5205, F-69622 Villeurbanne, France
Paul Robertson PAULR@DOLLABS.COM
DOLL Labs, Lexington, MA, USA

Editor: Editor's name

Abstract

We introduce the problem for a robot to localize itself, and, simultaneously, actively infer the existence and properties of *phenomena* present in its surrounding environment: the SLAPI problem. A phenomenon is a representation of an entity "as the robot experiences it" through interaction. The SLAPI problem relates to the SLAM problem but differs in that it does not aim at constructing a precise map of the environment, and it can apply to robots with coarse sensors. We demonstrate a SLAPI algorithm to control a robot equipped with omni-directional wheels, an echo-localization sensor, photosensitive sensors, and an inertial measurement unit, but no precise sensors like camera, lidar, or odometry. As the robot circles around an object, it constructs the phenomenon corresponding to this object under the form of the set of the spatially-localized control loops of interaction that the object affords to the robot. SLAPI algorithms could help design companion robots that mimic intrinsic motivation such as curiosity and playfulness. Further studies of the SLAPI problem could improve the scientific understanding of how cognitive beings construct knowledge about objects from sensorimotor experience of interaction.

Keywords: Constructivism; active inference; enaction; autonomous robotics; SLAM

1. Introduction

The problem of getting mobile robots to autonomously learn the position of surrounding objects, recognize them, and keep track of their relative displacements is considered by many to be a key prerequisite of truly autonomous robots. Within this framework, the SLAM problem (Simultaneous Localization and Mapping) has been formalized and studied since the 1990s: constructing and updating a map of an unknown environment while simultaneously keeping track of the robot's position within it (e.g., Taketomi et al., 2017). SLAM algorithms are tailored to the available resources: odometric sensors, sensors of the environment, computational capacities, as well as the landmarks' properties, quantity, and dynamics, and the usage intended for the robot.

When displacements are imprecise and odometric data is not available, when landmarks are not directly identifiable, and below a certain level of scarcity and noise in the sensory

data relative to the environment's complexity, it becomes difficult to perform SLAM accurately enough to use the robot for tasks involving complex navigation (Gay et al., 2021). For such robots, we propose the SLAPI problem: Simultaneous Localization and Active Phenomenon Inference. In contrast with SLAM, SLAPI does not aim at constructing a map to use for navigation. Instead, it aims at organizing behavior spatially in the vicinity of objects to design robots that mimic intrinsic motivation such as playfulness and curiosity as they discover and interact with unknown objects (e.g., Oudeyer et al., 2007). Possible applications may not include delivery tasks but may include entertainment and games with lifelike companion robots.

SLAPI makes no assumption that landmarks can be directly distinctively identified through sensors. The robot must rather actively interact with objects, possibly from different angles and through different modalities of control loops, to categorize and recognize objects, and possibly use them as landmarks. We call this process *active phenomenon inference*, in line with the theory of active inference (e.g., Friston et al., 2021).

2. The representational status of sensory data

An autonomous agent faces the necessity to actively infer the presence and the properties of objects in its environment when such presence and properties are not directly registered in sensory data. This raises the question of the *representational status of sensory* data: is sensory data representational or not? This question has been discussed time and again at the philosophical level (e.g., Williford, 2013). Loosely, two hypotheses collide: the hypothesis that sensory data carry information about features of the world, versus the hypothesis that sensory data carry information about the agent's experience of interaction with the world. We refer to the former as the *representationalist hypothesis*, and to the latter as the *constructivist hypothesis* because it relates to Piaget's theory of constructivist learning based on sensorimotor schemes (Guillermin and Georgeon, 2022).

SLAPI specifically helps investigate the constructivist hypothesis because it applies to robots that have coarse sensors that do not provide much descriptive information about the environment. The robot must probe the environment a little bit like a blind person who uses a cane to actively construct a mental representation of its surroundings. Probing experiences consist of control loops during which the robot interacts with the environment. They are triggered by an action selected by the robot and result in an outcome. The outcome is informative not of the object itself but of the possibility of interaction afforded by the object to the robot. Past probing experiences drive future behavior because they signal affordances for action. Figure 1 shows this cycle of interaction. The software selects an action associated with spatial information that specifies how the control loop should be enacted in the world. In return, the software receives an outcome associated with spatial information that describes how the control loop has been enacted depending on the actual nature and position of surrounding objects.

Note that the constructivist hypothesis accepts that the outcome may sometimes register features of the environment but avoids "baking" this assumption in the algorithm *a priori*. Rudrauf et al. (2017) have also proposed an active inference model related to the constructivist hypothesis. They state that "All we need here is the idea that in one way or

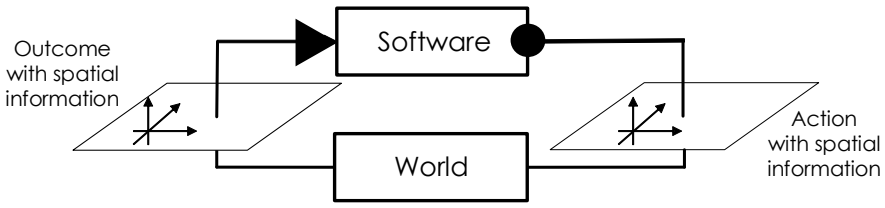

Figure 1: The interaction cycle. Black bullet: the cycle begins with the software selecting an action containing spatial information to enact in the world (right). Black arrowhead: the cycle ends with the software receiving the outcome containing spatial information (left).

another the sensory organs provide an independent source of input and correction for the continually updated world model" (p. 19).

3. The experimental setup

We use the robot cat of brand Osoyoo[1], to which we added an inertial measurement unit (Figure 2). We use the omni-directional wheels only to move longitudinally or laterally (sweep). Section 4.1 will explain the primitive moves. Each primitive move induces an incertitude of displacement of about $\pm 20\%$ that we can't measure. The inertial measurement unit can only measure the yaw with the relatively good accuracy of $\pm 1°$. It also provides a compass with an accuracy of $\pm 5°$. We compute the robot's azimuth (angle from North) using both yaw integration (for precision) and compass (to correct drift). We then devise the absolute head direction knowing the head angle relative to the robot with good precision.

The robot can turn its head to perform echo localization in different directions. The distance measure has an accuracy of $\pm 2mm$ but the detection cone spans 70°. We implemented two methods to estimate the direction of an object. Method 1 consists of scanning every 10° through the full range of head direction $[-\pi/2, \pi/2]$ and then computing the center of "strikes" of similar distances. Method 2 consists of turning the head by steps of 10° until finding a local minimum distance. We use Method 1 to find objects in the surrounding. We programmed the robot to keep its head aligned toward a *focus point* during its moves, and then to performs Method 2 to re-align its head towards the object.

We set up an environment with various objects that can be detected through echo-localization, and a black circle to delimit the robot's territory. With the inaccuracy of measures listed above, it is challenging to infer the position and shapes of objects. On the other hand, the robot has no target to reach. We only want it to explore its environment.

4. The software architecture and algorithm

As a proof of concept to illustrate the SLAPI problem, we implemented the cognitive architecture depicted in Figure 3. This architecture is implemented on a remote PC and takes

1. https://osoyoo.com/2019/11/08/omni-direction-mecanum-wheel-robotic-kit-v1/

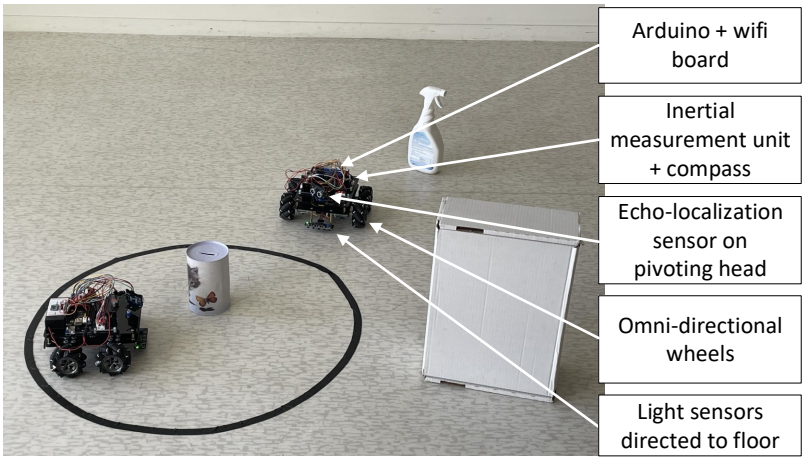

Figure 2: The experimental setup. Osoyoo robots, a black line on the floor, and various objects detectable through echo-localization. The omni-directional wheels allow the robot to translate laterally (sweep right or left).

the place of the *Software* in Figure 1. The robot plus the environment take the place of the *World*.

The architecture follows a regular Model-View-Controller design pattern in which the *Memory* plays the role of the database, and the *Workspace* plays the role of the Model. As we will further develop, the *Memory* contains the egocentric memory and the allocentric memory which are displayed on screen via the *View Controllers* (Figure 3, right). The *Workspace* contains the *Decider* which implements the robot's policy, and the *Integrator* which infers the phenomena.

4.1. The control loops of interaction implemented in the robot

A C++ program on the robot's Arduino board handles the reception of the action, drives the control loop, and then returns the outcome through wifi. Table 1 lists the supported actions and outcomes.

The reception of an action triggers the enaction of the corresponding control loop until its termination condition is satisfied. For example, the `Forward` action sets the robot in motion. It has three possible termination conditions: `Default`: time out of 1 second (approximately 20 cm traveled); `Line detection`: the floor sensor detects a line causing the robot to retreat back for a few centimeters; and `Impact`: the inertial measurement unit detects a strong deceleration indicating an impact with an object.

Besides the action code, the cognitive architecture sends two more values to the robot: the coordinates of the focus point if any, and an estimated speed. As introduced in Section 3, the robot uses them to keep its head aligned towards objects over successive moves. It also gives the human observer the impression that the robot keeps it attention on a particular object, making it look more alive.

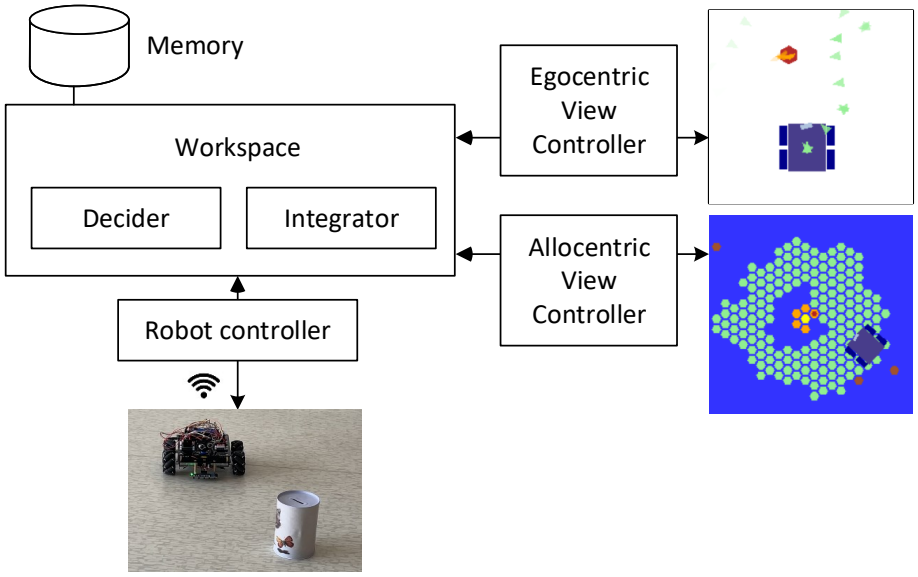

Figure 3: The software architecture implemented on a remote PC. Bottom: the robot recieves the action and sends back the outcome through wifi. Top-right: Egocentric memory. Bottom-right: Allocentric memory. Orange: echos, green: previous robot positions, red: focus point.

Table 1: Actions available to the robot and their possible outcomes

(a) Actions

Code	**Description**
`Forward`	During 1 sec.
`Backward`	or until
`Sweep left`	impact or line
`Sweep right`	detection
`Turn left`	$\pi/4$
`Turn right`	$-\pi/4$
`Head scan`	$[-\pi/2, \pi/2]$

(b) Outcomes

Code	**Description**
`Line left`	Floor sensors
`Line front`	cross luminosity
`Line right`	threshold
`Impact`	Violent deceleration
`Echo lost focus`	No echo where expected
`Echo left`	Direction and range of the nearest echo
`Echo right`	
`Echo far left`	
`Echo far right`	
`Echo far front`	
`Echo close front`	
`Default`	No line, no echo

The robot also returns additional information to the cognitive architecture: yaw, azimuth (angle relative to the north), and duration of the various phases of the control loop. The cognitive architecture uses this information to update the spatial memory based on the robot's displacement. The termination phase of the control loop aligns the robot's head towards the nearest echo. The robot then also returns the nearest echo measure along with the head direction. The cognitive architecture uses this information to mark the position of the echo in spatial memory. This will be used to infer the presence of an object when multiple echos are localized in the same area. Table 2 summarizes the data exchanged between the PC and the robot.

Table 2: Dialogue between the PC and the robot through wifi

PC to Robot	Action code, focus coordinates (x, y), estimated speed (x, y)
Robot to PC	Outcome code, echo distance, head direction, yaw, azimuth, duration

4.2. Egocentric memory

The egocentric memory is a short-term spatial memory of the experiences of interaction in the surrounding of the robot. It is inspired by the brain's egocentric cells located in the superior colliculus (Grieves and Jeffery, 2017).

Technically, it stores the robot's *experiences* in a coordinate system centered on the robot. An experience is a data structure that contains the action, the outcome, the position in space relative to the robot, the timestamp. Our implementation initializes the position from a hard-coded model of the robot: line-detection experiences are placed at the position of the floor sensors; echo experiences are placed at the estimated origin of the echo using head direction and measure of echo distance. Additionally, echo experiences also store the direction of the sensor needed by the phenomenon inference function. When the robot moves, the positions of experiences are then moved opposite by the robot's estimated displacement (Figure 3, top right). Of course, errors in the estimation of the robot's position accumulate causing a drift in egocentric memory. In our experiment, we observed that the position of experiences were unreliable after 5 to 10 interaction cycles.

4.3. Allocentric memory

The allocentric memory is inspired by *grid cells* in the entorhinal cortex (Grieves and Jeffery, 2017), and reproduces their hexagonal structure. Technically, it stores a set of *affordances* that represent the possibilities of interaction afforded by the environment to the robot. An affordance is an experience associated with its position in allocentric reference (Figure 4).

Converting from egocentric to allocentric reference requires establishing an origin point from which path integration can be performed. If this origin point is immobile and recognizable later, it can be used to correct the robot's position and avoid infinite accumulation of position errors. Fortunately, in our case, we can use the position of the echo associated with the absolute head direction to recognize the origin point when the robot has performed

a complete tour around the object. This solution works if the object is convex, and as long as the robot does not mistake objects.

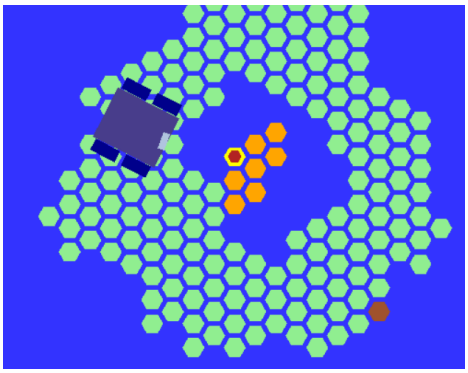

Figure 4: Example allocentric memory. Green: cells previously traversed by the robot. Orange: echos localized with Method 1. Yellow: origin of the phenomenon. Small red hexagon: current focus point. Brown: echos localized with method 2 (in this example coming from another object).

4.4. The phenomenon integrator

The phenomenon integrator constructs a data structures that represents an object in term of affordances. We call such a data structure a *phenomenon* in compliance with the common sense usage of this term: the perception by a cognitive being of "something" in the environment. (Thórisson, 2021, p. 8) provides a more technical definition that also matches our usage: "any useful grouping of a subset of spatio-temporal patterns experienced by an agent in an environment". Notably, these action-related representations of objects also resemble *action codes* that have been found in parietal regions of the cerebral cortex of humans and other primates (Chao and Martin, 2000; Colby and Goldberg, 1999; Schubotz et al., 2014). Like our affordances, action codes are linked to memory traces of previous actions involving the object.

Our phenomenon data structure contains a set of affordances (Figure 5) and a *confidence coefficient* in the range $[0, 1]$. We implemented Algorithm 1 to address the difficulty of localizing affordances in the referential of an initially unknown phenomenon while the robot's estimated position is prone to error.

Algorithm 1 takes a newly found affordance in input, and returns the distance by which the robot's position must be adjusted. It compares the position of the new affordance with the position of the nearest affordance already attached to this phenomenon. If the difference is below *MAX_DISTANCE*, it assumes that the new affordance belongs to this phenomenon. The method *similar_to()* compares the new affordance with the *origin affordance* (the first affordance attached to this phenomenon). This comparison is based on the affordance's position and on the absolute head direction. The method *increase_confidence()* keeps track of the number of tours the robot has made around the object. When a new affordance is

found similar to the origin affordance after a new tour, then the phenomenon's confidence coefficient is increased. The *position_adjustment* vector is computed to reposition the robot at the origin position relative to the *origin affordance*. When the new affordances is not similar to the *origin affordance*, the length of the *position_adjustment* vector is proportional to the confidence coefficient. Both the position of the new affordance and the position of the robot are adjusted by the adjustment vector. At the beginning, when the confidence is close to zero, the robot rests on its estimated position to roughly estimate the phenomenon's shape. When the confidence becomes close to one after several tours, the robot ceases updating the phenomenon's shape and rather updates its own position with regard to the object based on the distance measured through echo-localization. *Phenomenon.prune()* removes old affordances near the new one that are assumed to have become irrelevant. *Phenomenon.append()* attach the new affordance to the phenomenon.

Algorithm 1: Phenomenon.update(affordance)

$nearest\_affordance \leftarrow phenomenon.find\_nearest(affordance)$;
$delta \leftarrow nearest\_affordance.position - affordance.position$;
if $delta < MAX\_DISTANCE$ **then**
    if $affordance.similar\_to(origin\_affordance)$ **then**
        $phenomenon.increase\_confidence()$;
        $position\_adjustment = origin\_affordance.position$ - affordance.position;
    else
        $position\_adjustment = phenomenon.confidence * delta$;
        $affordance.position \leftarrow affordance.position + position\_adjustment$;
        $phenomenone.prune(affordance)$;
        $phenomenon.append(affordance)$;
    end
end
return *position_adjustment*

4.5. The decider

The Decider implements the policy that selects the robot's actions. For our proof of concept, we used an algorithm inspired by our previous work (Georgeon et al., 2013; Robertson and Laddaga, 2009). It stores a *list of behaviors* in the form of *sequences of experiences* that the robot can enact. After each interaction cycle, the Decider activates the sequences in this list whose beginning sub-sequence matches experiences present in memory. The activated sequences then propose their ending sub-sequences for further enaction with a weight. Finally the Decider selects the sub-sequence that is proposed with the highest weight. We have initialized the list of behaviors with predefined behavioral patterns that give the robot a tendency to circle around objects when it detects them.

With this policy, the robot wanders randomly until it detects an object nearby, and then it circles around this object until the experimenter removes the object. This is sufficient for our demonstration. More complex information-seeking policies could be used. Gottlieb and Oudeyer (2018) provide a recent literature review in this topic.

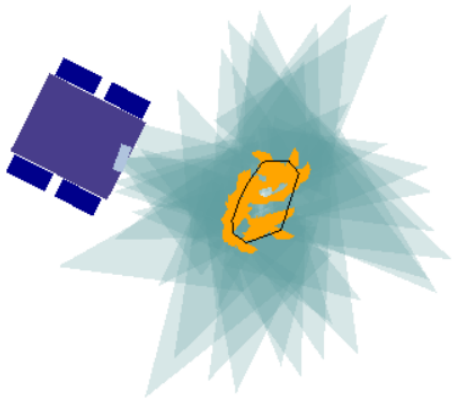

Figure 5: Example phenomenon. Gray triangles and orange half-circles represent the affordances experienced by the robot from different positions and attached to this phenomenon. Gray triangles are the cones of echo-localization. Orange half-circles are the estimated positions of the echos. The object was made of two cans similar to the one in Figure 3 next to each other. Its estimated shape is outlined.

5. Analysis

Figure 6 reports two different phenomena constructed from two kinds of object: an object that affords echo and impact (top), and an object that also affords line detection (bottom). A representative run can be seen in video (Knockaert, 2022). It shows the robot circling around the object while constructing the egocentric and allocentric maps. It is worth noting that many human observers attribute the intention of the robot to carefully observe the object due to the fact that it keeps its head pointing to the object.

6. Conclusion

This paper presents the SLAPI problem: the problem for an autonomous artificial agent to construct knowledge about entities in its environment from sensorimotor experience of interaction. This problem is particularly salient when the agent has rudimentary sensors that provide poor information about the features of the environment. In this case, to be informative, sensory data should rather be considered as *outcome* of a control loop than *percepts*.

On the contrary, when the agent has rich sensors that allow the identification of features of the world, it is tempting for the developer of the agent's software to consider that sensory data carries representational information about the world. Philosophy of mind, however, provides theoretical arguments against this representationalist hypothesis. These arguments go back to Kant, who claimed that the world "in itself" is unknowable, and relate to Piaget's developmental psychology and theory of enaction (e.g., Froese and Ziemke, 2009). If we trust these arguments, we can expect SLAPI algorithms to also bring value to agents that have rich sensors. They could endow such agents with more autonomy in the way they construct

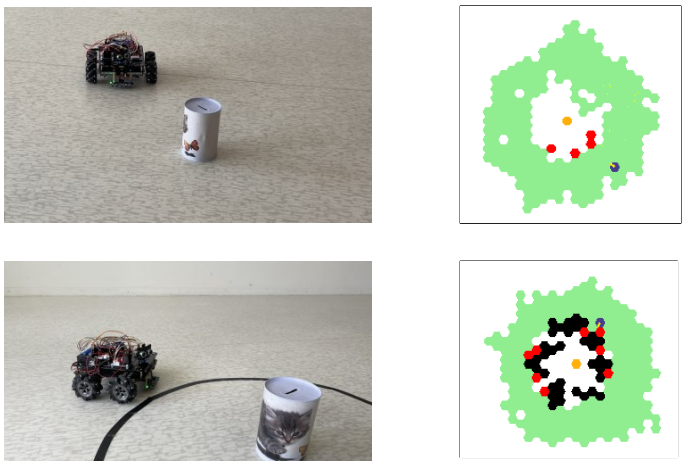

Figure 6: Examples of phenomenon inference. Left: the robot and the object seen by the human observer. Right: the phenomenon constructed by the robot. Top: the object affords echo (orange cells) and impact (red cells). Bottom: the object also affords line detection (black cells). Green cells represent empty space.

their own knowledge of the world, and keep this knowledge grounded in their individual experience of interaction.

We reported our implementation and experiment as an initial example to illustrate the SLAPI problem, in the hope that it provides clarification. We hope that future models will address more complex problems of phenomenal inference, such as categorization and recognition of similarities and differences between types of phenomena. A more precise evaluation of their performances may also tell us whether they hold further similarities with observed neuronal representations of action in the brain. More broadly, such models can improve our understanding of how animals construct knowledge of objects through enaction.

Acknowledgments

We thank Célia Vaz-Cerniglia, Brigitte Blanquet, and Robin Couture for their support to conduct the robotics experiment.

References

L. L. Chao and A. Martin. Representation of manipulable man-made objects in the dorsal stream. *NeuroImage*, 12(4):478–484, October 2000. ISSN 1053-8119. doi: 10.1006/nimg.2000.0635.

C. L. Colby and M. E. Goldberg. Space and attention in parietal cortex. *Annual Review of Neuroscience*, 22:319–349, 1999. ISSN 0147-006X. doi: 10.1146/annurev.neuro.22.1.319.

Karl Friston, Rosalyn J. Moran, Yukie Nagai, Tadahiro Taniguchi, Hiroaki Gomi, and Josh Tenenbaum. World model learning and inference. *Neural Networks*, 144:573–590, December 2021. ISSN 0893-6080. doi: 10.1016/j.neunet.2021.09.011. URL https://www.sciencedirect.com/science/article/pii/S0893608021003610.

Tom Froese and Tom Ziemke. Enactive artificial intelligence: Investigating the systemic organization of life and mind. *Artificial Intelligence*, 173(3-4):466–500, March 2009. ISSN 00043702. doi: 10.1016/j.artint.2008.12.001. URL https://linkinghub.elsevier.com/retrieve/pii/S0004370208002105.

Simon Gay, Kévin Le Run, Edwige Pissaloux, Katerine Romeo, and Christèle Lecomte. Towards a predictive bio-inspired navigation model. *Information*, 12(3):100, March 2021. ISSN 2078-2489. doi: 10.3390/info12030100. URL https://www.mdpi.com/2078-2489/12/3/100.

Olivier L. Georgeon, James B. Marshall, and Riccardo Manzotti. ECA: An enactivist cognitive architecture based on sensorimotor modeling. *Biologically Inspired Cognitive Architectures*, 6:46–57, October 2013. ISSN 2212683X. doi: 10.1016/j.bica.2013.05.006. URL https://linkinghub.elsevier.com/retrieve/pii/S2212683X13000418.

Jacqueline Gottlieb and Pierre-Yves Oudeyer. Towards a neuroscience of active sampling and curiosity. *Nature Reviews. Neuroscience*, 19(12):758–770, December 2018. ISSN 1471-0048. doi: 10.1038/s41583-018-0078-0.

Roddy M. Grieves and Kate J. Jeffery. The representation of space in the brain. *Behavioural Processes*, 135:113–131, 2017. ISSN 0376-6357. doi: https://doi.org/10.1016/j.beproc.2016.12.012. URL https://www.sciencedirect.com/science/article/pii/S0376635716302480.

Mathieu Guillermin and Olivier Georgeon. Artificial Interactionism: Avoiding isolating perception from cognition in AI. *Frontiers in Artificial Intelligence*, 5, 2022. ISSN 2624-8212. doi: 10.3389/frai.2022.806041. URL https://www.frontiersin.org/article/10.3389/frai.2022.806041.

Titouan Knockaert. Demonstration of phenomenon inference, July 2022. URL https://youtu.be/Sue9yMDq0E8.

Pierre-Yves Oudeyer, Frdric Kaplan, and Verena V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11(2):265–286, April 2007. ISSN 1089-778X. doi: 10.1109/TEVC.2006.890271. URL http://ieeexplore.ieee.org/document/4141061/.

P. Robertson and R. Laddaga. A biologically inspired spatial computer that learns to see and act. In *Spatial Computing Workshop, SASO 2009, San Francisco*, 2009.

David Rudrauf, Daniel Bennequin, Isabela Granic, Gregory Landini, Karl Friston, and Kenneth Williford. A mathematical model of embodied consciousness. *Journal of Theoretical Biology*, 428:106–131, September 2017. ISSN 00225193. doi: 10.1016/j.jtbi.2017.05.032. URL https://linkinghub.elsevier.com/retrieve/pii/S0022519317302540.

Ricarda I. Schubotz, Moritz F. Wurm, Marco K. Wittmann, and D. Yves von Cramon. Objects tell us what action we can expect: dissociating brain areas for retrieval and exploitation of action knowledge during action observation in fMRI. *Frontiers in Psychology*, 5:636, June 2014. ISSN 1664-1078. doi: 10.3389/fpsyg.2014.00636. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4067566/.

Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. *Visual SLAM algorithms: a survey from 2010 to 2016*. 2017. Publication Title: IPSJ Transactions on Computer Vision and Applications.

Kristinn R. Thórisson. *The 'Explanation Hypothesis' in general self-supervised Learning*. International Workshop in Self-Supervised Learning, 2021.

Kenneth Williford. Husserl's hyletic data and phenomenal consciousness. *Phenomenology and the Cognitive Sciences*, 12(3):501–519, 2013. doi: 10.1007/s11097-013-9297-z. Publisher: Springer Verlag.