Simultaneous Localization and Active Phenomenon Inference (SLAPI)

Olivier L. Georgeon OGEORGEON@UNIV-CATHOLYON.FR
UR Confluence, Sciences et Humanits (EA 1598) - Lyon Catholic University, France
Titouan Knockaert TITOUAN.KNOCKAERT@GMAIL.COM
Universit Claude Bernard Lyon 1, LIRIS CNRS UMR5205, F-69622 Villeurbanne, France
Juan R. Vidal JVIDAL@UNIV-CATHOLYON.FR
UR Confluence, Sciences et Humanits (EA 1598) - Lyon Catholic University, France

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 pet robots that mimic intrinsic motivation such as curiosity and playfulness with toys. 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: Constructivist learning; active inference; 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., **?**). 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 environments complexity, it becomes difficult to perform SLAM accurately enough to use the robot for tasks involving complex navigation (**?**). 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., **?**). Possible applications may not include delivery tasks but may include entertainment and games, similar to playing with pets.

SLAPI makes no assumption that landmarks can be directly and passively uniquely 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., **?**).

2. The representational status of sensory data

An autonomous agent faces the necessity to actively infer the existence and the properties of objects in its environment when such existence 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., **?**). 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 (**?**).

SLAPI falls within 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 constitute 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*. **?** have also proposed an active inference model related to the constructivist hypothesis. He states that "All we need here is the idea that in one way or another the sensory organs provide an independent source of input and correction for the continually updated world model" (p. 19).

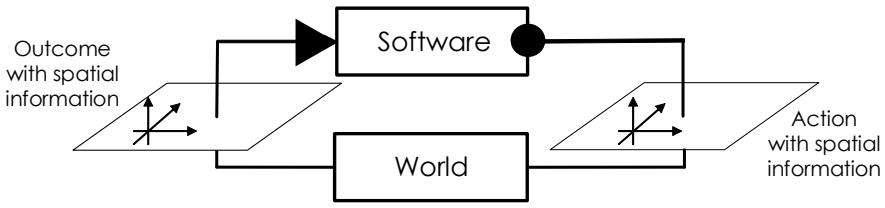

Figure 1: The interaction cycle. Black bullet: the cycle begins with the software selecting an action containing spatial information to enact in the world (right). Black arrowhead: the cycle ends with the software receiving the outcome containing spatial information (left).

3. The experimental setup

We use the robot cat of brand Osoyoo[1], to which we added an inertial measurement unit (Figure 2). We set up an environment with black lines on the floor and various objects that can be detected through the echo-localization sensor. There is not a final target to reach. We only want the robot to explore its environment.

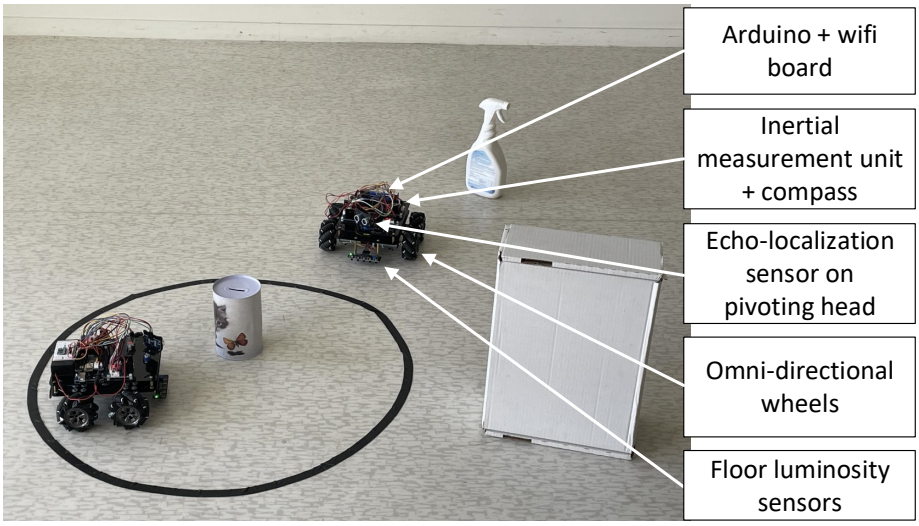

Figure 2: The experimental setup. Osoyoo robots, a black line on the floor, and various objects detectable through echo-localization. The omni-directional wheels allow the robot to translate laterally (sweep right or left).

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

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 the place of the *Software* in Figure 1. The robot plus its environment takes 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.

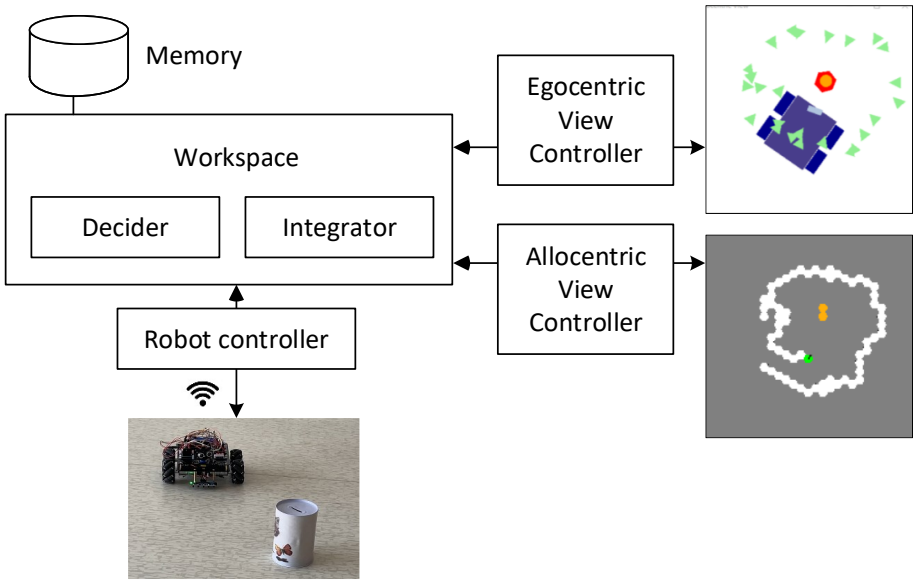

Figure 3: The software architecture implemented on a remote PC. Bottom: the robot recieves the action and sends back the outcome through wifi. Top-right: Egocentric memory shows the position of echos (orange circles), the focus of attention (red hexagon), and the previous positions of the robot (green triangles). Bottom-right: Allocentric memory shows the position of the echo (orange cells), the explored cells (white), the position of the robot (green), the unexpored cells (grey).

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.

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

(a) Actions

Code	**Description**
`Forward`	During 1 sec.
`Backward`	or until shock
`Sweep left`	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
`Shock`	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 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 `Shock`: 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 a focus point, and an estimated speed. The robot's program uses these to keep the robot's head aligned towards the focus point during the robot's displacement. This helps the robot keep track of objects over various moves. It also gives the human observer the impression that the robot keeps it attention on a particular object, making it look more alive.

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. The egocentric memory

The egocentric memory is a short-term 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 (**?**).

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, and 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. 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 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. The allocentric memory

The allocentric memory stores a spatial representation in a coordinate system relative the environment. It is inspired by *grid cells* in the hippocampus of the mammalian brain (**?**).

Our implementation reproduces the grid cell's hexagonal structure (Figure 3, bottom right). Each cell stores the list of experiences enacted in the location referenced by the cell, and can have one of the following status:

Unknown No experience has been attempted here.

Occupied The robot is localized here.

Empty The robot has been here in the past.

Experience One or several experiences have been localized here.

Allocentric memory is used to construct a local spatial representation of the objects with which the robot is interacting. It keeps track of the robot's position through path integration. It informs the next action to select by the *Decider*, and constitutes the base for phenomenal inference performed by the *Integrator*.

4.4. The decider

The Decider implements the policy that selects the robot's actions. For our proof of concept, we used an algorithm we designed previously (**?**). 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. **?** provide a recent literature review in this regard.

4.5. The phenomenon integrator

We called this process Integrator because it integrates a heterogeneous set of experiences into a single data structure that represents the object that affords these experiences. We called this 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. **?**'s paper 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".

Once the robot has circled around an abject, the object becomes identified as a single entity, then the Integrator copies its representation from allocentric memory to the phenomenon data structure. The phenomenon thus stores a local map of experiences afforded by the object.

Notably, these action-related representations of objects resemble *action codes* that have been found in parietal regions of the cerebral cortex of humans and other primates (**???**). Like our sequences of experiences, action codes are linked to memory traces of previous actions involving the object.

5. Results

Figure 4 reports two different phenomena constructed from two kinds of object: an object that affords echo and shock (top), and an object that also affords line detection (bottom). A representative run can be seen in video (**?**). 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*.

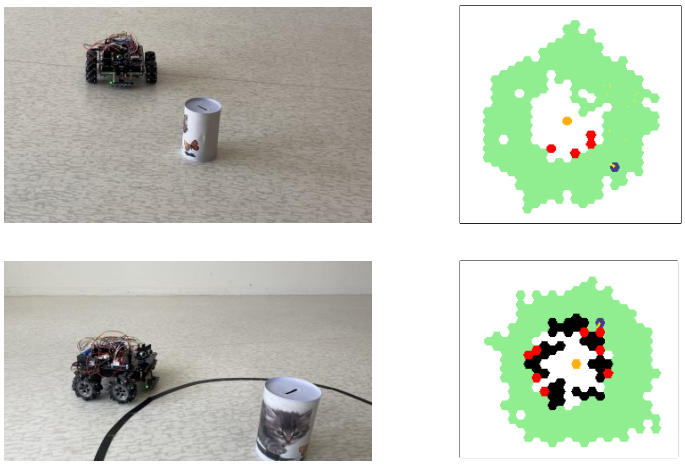

Figure 4: Examples of phenomenon inference. Left: the robot and the object seen by the human observer. Right: the phenomenon constructed by the robot. Top: the object affords echo (orange cells) and shock (red cells). Bottom: the object also affords line detection (black cells). Green cells represent empty space.

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, provide 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 constructivist epistemology. They also have echos in modern physics. 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 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 experience of interaction.