

Simultaneous Localization and Active Phenomenon Inference (SLAPI)

Olivier L. Georgeon

OGEORGEON@UNIV-CATHOLYON.FR

UR Confluence, Sciences et Humanités (EA 1598) - Lyon Catholic University, France

Titouan Knockaert

TITOUAN.KNOCKAERT@GMAIL.COM

Université Claude Bernard Lyon 1, LIRIS CNRS UMR5205, F-69622 Villeurbanne, France

Editor: Editor's name

Abstract

This is the abstract for this article.

Keywords: List of keywords

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. 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. 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 exhibit intrinsic motivation such as playfulness and curiosity as they discover and interact with unknown objects. 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. This process is called active phenomenon inference. Here the term phenomenon refers to the knowledge of physical objects actively constructed by the robot from its point of view and "as the robot experiences the object through interaction". We designed a proof-of-concept algorithm to illustrate the SLAPI problem. We demonstrated it in a robot mounted on omnidirectional wheels and endowed with an echo-localization sensor, photosensitive sensors, and an inertial measurement unit, but no camera

and lidar. 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 that the object affords to the robot. New elements can be subsequently added to this set as the robot improves its knowledge of the object. Results show that the robot drew out a few cries of amusement and endearment from some human observers, which encourages us to keep improving this range of algorithms. Moreover, we believe that the study of SLAPI problems can shed some light on how animals construct knowledge of objects through sensorimotor interactions, while keeping this knowledge grounded in experience. It can also provide an angle of attack to the more general AI problem of self-motivated open-ended learning in the real world.

NOTE:

This is an numbered theorem-like environment that was defined in this document's preamble.

1.1. Sub-sections

Sub-sections are produced using `\subsection`.

1.1.1. SUB-SUB-SECTIONS

Sub-sub-sections are produced using `\subsubsection`.

Sub-sub-sub-sections Sub-sub-sub-sections are produced using `\paragraph`. These are unnumbered with a running head.

Sub-sub-sub-sub-sections Sub-sub-sub-sub-sections are produced using `\subparagraph`. These are unnumbered with a running head.

2. Cross-Referencing

Always use `\label` and `\ref` (or one of the commands described below) when cross-referencing. For example, the next section is Section 3. The `jmlr` class provides some convenient cross-referencing commands: `\sectionref`, `\equationref`, `\tableref`, `\figureref`, `\algorithmref`, `\theoremref`, `\lemmaref`, `\remarkref`, `\corollaryref`, `\definitionref`, `\conjectureref`, `\axiomref`, `\exampleref` and `\appendixref`. The argument of these commands may either be a single label or a comma-separated list of labels. Examples:

Referencing sections: Section 3 or Sections 1 and 3 or Sections 1, 3, 5.1 and 5.2.

Referencing equations: Equation (1) or Equations (1) and (3) or Equations (1), (2), (3) and (4).

Referencing tables: Table 1 or Tables 1 and ?? or Tables 1, ?? and ??.

Referencing figures: Figure ?? or Figures ?? and 4 or Figures ??, 4 and 5 or Figures 5(a) and 5(b).

Referencing algorithms: Algorithm 1 or Algorithms 1 and 2 or Algorithms 1, 2 and 3.

Referencing theorem-like environments: Theorem 1, Lemma 2, Remark 3, Corollary 4, Definition 5, Conjecture 6, Axiom 7 and Example 1.

Referencing appendices: Appendix A or Appendices A and B.

3. Equations

The `jmlr` class loads the `amsmath` package, so you can use any of the commands and environments defined there. (See the `amsmath` documentation for further details.¹)

Unnumbered single-lined equations should be displayed using `\[` and `\]`. For example:

$$E = mc^2$$

Numbered single-line equations should be displayed using the `equation` environment. For example:

$$\cos^2 \theta + \sin^2 \theta \equiv 1 \tag{1}$$

This can be referenced using `\label` and `\equationref`. For example, Equation (1).

Multi-lined numbered equations should be displayed using the `align` environment.² For example:

$$f(x) = x^2 + x \tag{2}$$

$$f'(x) = 2x + 1 \tag{3}$$

Unnumbered multi-lined equations should be displayed using the `align*` environment. For example:

$$\begin{aligned} f(x) &= (x+1)(x-1) \\ &= x^2 - 1 \end{aligned}$$

If you want to mix numbered with unnumbered lines use the `align` environment and suppress unwanted line numbers with `\nonumber`. For example:

$$\begin{aligned} y &= x^2 + 3x - 2x + 1 \\ &= x^2 + x + 1 \end{aligned} \tag{4}$$

An equation that is too long to fit on a single line can be displayed using the `split` environment. Text can be embedded in an equation using `\text` or `\intertext` (as used in Theorem 1). See the `amsmath` documentation for further details.

3.1. Operator Names

Predefined operator names are listed in Table 1. For additional operators, either use `\operatorname`, for example `\operatorname{var}(X)` or declare it with `\DeclareMathOperator`, for example

```
\DeclareMathOperator{\var}{var}
```

and then use this new command. If you want limits that go above and below the operator (like `\sum`) use the starred versions (`\operatorname*` or `\DeclareMathOperator*`).

1. Either `texdoc amsmath` or <http://www.ctan.org/pkg/amsmath>

2. For reasons why you shouldn't use the obsolete `eqnarray` environment, see Lars Madsen, *Avoid eqnarray!* TUGboat 33(1):21–25, 2012.

Table 1: Predefined Operator Names (taken from `amsmath` documentation)

<code>\arccos</code>	<code>arccos</code>	<code>\deg</code>	<code>deg</code>	<code>\lg</code>	<code>lg</code>	<code>\projlim</code>	<code>projlim</code>
<code>\arcsin</code>	<code>arcsin</code>	<code>\det</code>	<code>det</code>	<code>\lim</code>	<code>lim</code>	<code>\sec</code>	<code>sec</code>
<code>\arctan</code>	<code>arctan</code>	<code>\dim</code>	<code>dim</code>	<code>\liminf</code>	<code>lim inf</code>	<code>\sin</code>	<code>sin</code>
<code>\arg</code>	<code>arg</code>	<code>\exp</code>	<code>exp</code>	<code>\limsup</code>	<code>lim sup</code>	<code>\sinh</code>	<code>sinh</code>
<code>\cos</code>	<code>cos</code>	<code>\gcd</code>	<code>gcd</code>	<code>\ln</code>	<code>ln</code>	<code>\sup</code>	<code>sup</code>
<code>\cosh</code>	<code>cosh</code>	<code>\hom</code>	<code>hom</code>	<code>\log</code>	<code>log</code>	<code>\tan</code>	<code>tan</code>
<code>\cot</code>	<code>cot</code>	<code>\inf</code>	<code>inf</code>	<code>\max</code>	<code>max</code>	<code>\tanh</code>	<code>tanh</code>
<code>\coth</code>	<code>coth</code>	<code>\injlim</code>	<code>injlim</code>	<code>\min</code>	<code>min</code>		
<code>\csc</code>	<code>csc</code>	<code>\ker</code>	<code>ker</code>	<code>\Pr</code>	<code>Pr</code>		
		<code>\varlimsup</code>	$\overline{\lim}$	<code>\varinjlim</code>	\varinjlim		
		<code>\varliminf</code>	$\underline{\lim}$	<code>\varprojlim</code>	\varprojlim		

4. Vectors and Sets

Vectors should be typeset using `\vec`. For example \mathbf{x} . The `jmlr` class also provides `\set` to typeset a set. For example \mathcal{S} .

5. Floats

Floats, such as figures, tables and algorithms, are moving objects and are supposed to float to the nearest convenient location. Please don't force them to go in a particular place. In general it's best to use the `htbp` specifier and don't put the figure or table in the middle of a paragraph (that is make sure there's a paragraph break above and below the float). Floats are supposed to have a little extra space above and below them to make them stand out from the rest of the text. This extra spacing is put in automatically and shouldn't need modifying.

To ensure consistency, please *don't* try changing the format of the caption by doing something like:

```
\caption{\textit{A Sample Caption.}}
```

or

```
\caption{\em A Sample Caption.}
```

You can, of course, change the font for individual words or phrases, for example:

```
\caption{A Sample Caption With Some \emph{Emphasized Words}.}
```

5.1. Tables

5.2. Figures

Figures should go in the `figure` environment. Within this environment, use `\floatconts` to correctly position the caption and center the image. Use `\includegraphics` for external

Table 2: Data exchanged between the PC and the robot through wifi

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

Table 3: Actions available to the robot and their possible outcomes

<i>(a)</i> Actions		<i>(b)</i> Outcomes	
Action	Span	Outcome	Description
Forward	During 1 second or until line detection	Line left	Floor sensors cross luminosity threshold
Backward		Line front	
Shift left		Line right	
Shift right		Echo lost focus	No echo where expected
Turn left	$\pi/4$	Echo left	Direction and range of the nearest echo
Turn right	$-\pi/4$	Echo right	
Head scan	$[-\pi/2, \pi/2]$	Echo far left	
		Echo far right	
		Echo far front	
		Echo close front	
		Default	No line, no echo

graphics files but omit the file extension. Do not use `\epsfig` or `\psfig`. If you want to scale the image, it's better to use a fraction of the line width rather than an explicit length. For example, see Figure ??.

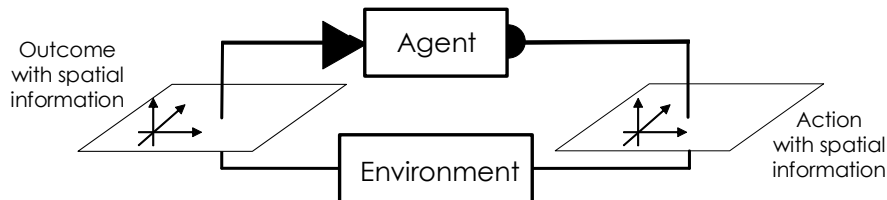


Figure 1: The interaction cycle

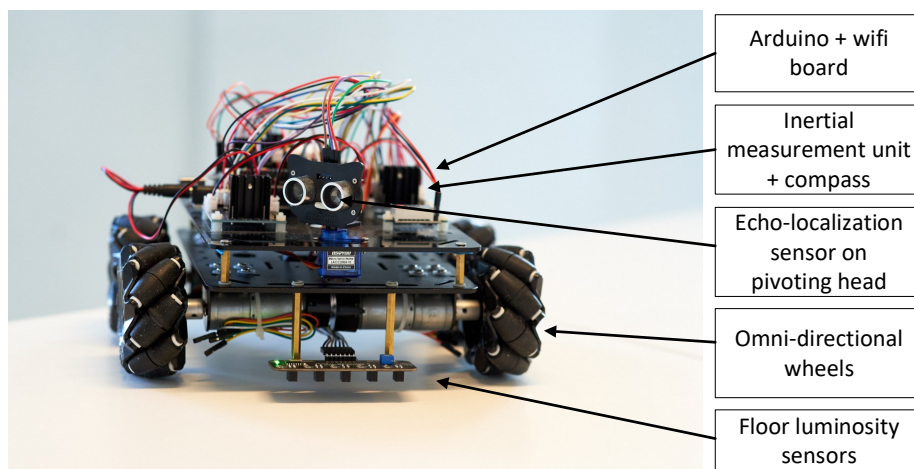


Figure 2: The robot

If your image is made up of \LaTeX code (for example, commands provided by the `pgf` package) you can include it using `\includeteximage` (defined by the `jmlr` class). This can be scaled and rotated in the same way as `\includegraphics`. For example, see Figure 4.

If the figure is too wide to fit on the page, you can use the `sidewaysfigure` environment defined in the `rotating` package.

It's best not to use `\graphicspath` with the `jmlr` class as it can cause problems with the production editing process. If the images are contained in a subdirectory, specify this when you include the image, for example `\includegraphics{figures/mypic}`.

5.3. Sub-Figures

Sub-figures can be created using `\subfigure`, which is defined by the `jmlr` class. The optional argument allows you to provide a subcaption. The label should be placed in the mandatory argument of `\subfigure`. You can reference the entire figure, for example Figure 5, or you can reference part of the figure using `\figureref`, for example Figure 5(a).

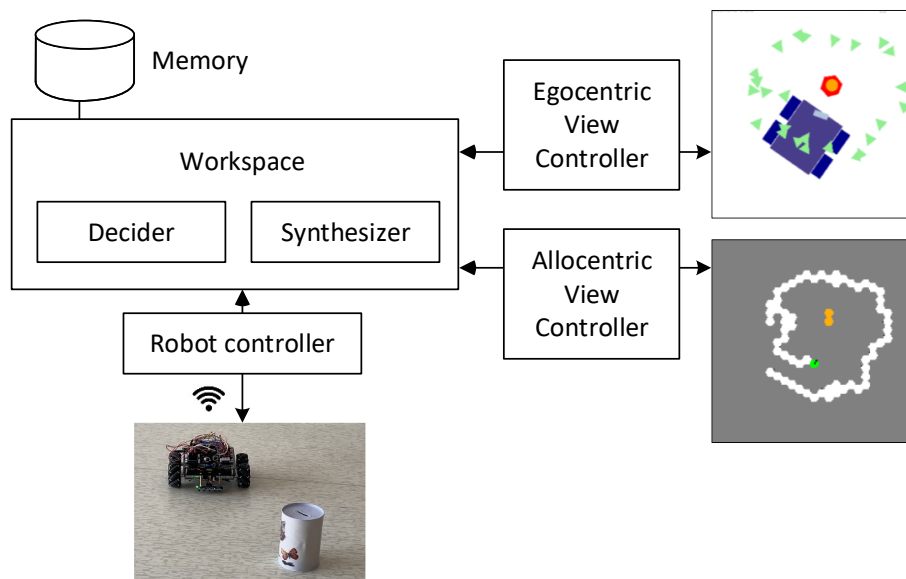


Figure 3: The Model-View-Controller architecture

Figure 4: The robot

Alternatively you can reference the subfigure using `\subfigref`, for example [\(a\)](#) and [\(b\)](#) in [Figure 5](#).

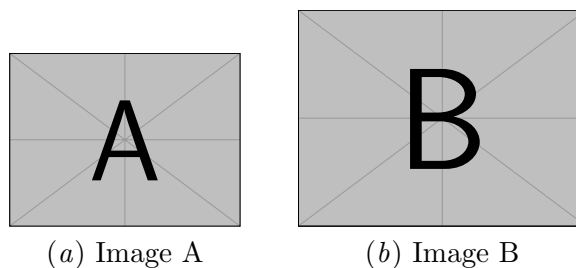


Figure 5: An Example With Sub-Figures.

By default, the sub-figures are aligned on the baseline. This can be changed using the second optional argument of `\subfigure`. This may be `t` (top), `c` (centered) or `b` (bottom). For example, the subfigures [\(a\)](#) and [\(b\)](#) in [Figure 6](#) both have `[c]` as the second optional argument.

Note that a very narrow sub-float will leave little space for the sub-caption, which will likely cause Underfull/Overfull hbox warnings.

You can set the length `\jmlrminsubcaptionwidth` to indicate the minimum width to be made available for the sub-captions, as in [Figure 7](#).

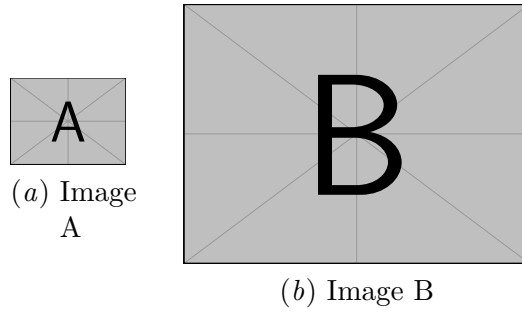


Figure 6: Another Example With Sub-Figures (with a Cramped Sub-Caption).

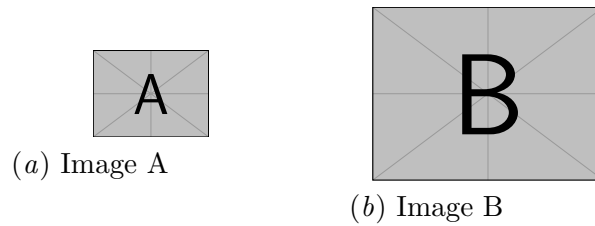


Figure 7: Another Example With Sub-Figures (with a Less Cramped Sub-Caption).

5.4. Algorithms

Enumerated textual algorithms can be displayed using the `algorithm` environment. Within this environment, use use an `enumerate` or nested `enumerate` environments. For example, see Algorithm 1. Note that algorithms float like figures and tables.

Algorithm 1: The Gauss-Seidel Algorithm

1. For $k = 1$ to maximum number of iterations

(a) For $i = 1$ to n

i. $x_i^{(k)} = \frac{b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^n a_{ij}x_j^{(k-1)}}{a_{ii}}$

ii. If $\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| < \epsilon$, where ϵ is a specified stopping criteria, stop.

If you'd rather have the same numbering throughout the algorithm but still want the convenient indentation of nested `enumerate` environments, you can use the `enumerate*` environment provided by the `jmlr` class. For example, see Algorithm 2.

Pseudo code can be displayed using the `algorithm2e` environment. This is defined by the `algorithm2e` package (which is automatically loaded) so check the `algorithm2e` documentation for further details.³ For an example, see Algorithm 3.

3. Either `texdoc algorithm2e` or <http://www.ctan.org/pkg/algorithm2e>

Algorithm 2: Moore's Shortest Path

Given a connected graph G , where the length of each edge is 1:

1. Set the label of vertex s to 0
 2. Set $i = 0$
 3. Locate all unlabelled vertices adjacent to a vertex labelled i and label them $i + 1$
 4. If vertex t has been labelled,

the shortest path can be found by backtracking, and the length is given by the label of t .

 otherwise

increment i and return to step 3
-

Algorithm 3: Computing Net Activation

Input: $x_1, \dots, x_n, w_1, \dots, w_n$

Output: y , the net activation

```

 $y \leftarrow 0;$ 
for  $i \leftarrow 1$  to  $n$  do
  |  $y \leftarrow y + w_i * x_i;$ 
end

```

6. Description Lists

The `jmlr` class also provides a description-like environment called `altdescription`. This has an argument that should be the widest label in the list. Compare:

add A method that adds two variables.

differentiate A method that differentiates a function.

with

add A method that adds two variables.

differentiate A method that differentiates a function.

7. Theorems, Lemmas etc

The following theorem-like environments are predefined by the `jmlr` class: `theorem`, `example`, `lemma`, `proposition`, `remark`, `corollary`, `definition`, `conjecture` and `axiom`. You can use the `proof` environment to display the proof if need be, as in Theorem 1.

Theorem 1 (Eigenvalue Powers) *If λ is an eigenvalue of B with eigenvector ξ , then λ^n is an eigenvalue of B^n with eigenvector ξ .*

Proof *Let λ be an eigenvalue of B with eigenvector ξ , then*

$$B\xi = \lambda\xi$$

premultiply by B :

$$\begin{aligned} BB\xi &= B\lambda\xi \\ \Rightarrow B^2\xi &= \lambda B\xi \\ &= \lambda\lambda\xi && \text{since } B\xi = \lambda\xi \\ &= \lambda^2\xi \end{aligned}$$

Therefore true for $n = 2$. Now assume true for $n = k$:

$$B^k\xi = \lambda^k\xi$$

premultiply by B :

$$\begin{aligned} BB^k\xi &= B\lambda^k\xi \\ \Rightarrow B^{k+1}\xi &= \lambda^k B\xi \\ &= \lambda^k\lambda\xi && \text{since } B\xi = \lambda\xi \\ &= \lambda^{k+1}\xi \end{aligned}$$

Therefore true for $n = k + 1$. Therefore, by induction, true for all n . ■

Lemma 2 (A Sample Lemma) *This is a lemma.*

Remark 3 (A Sample Remark) *This is a remark.*

Corollary 4 (A Sample Corollary) *This is a corollary.*

Definition 5 (A Sample Definition) *This is a definition.*

Conjecture 6 (A Sample Conjecture) *This is a conjecture.*

Axiom 7 (A Sample Axiom) *This is an axiom.*

Example 1 (An Example) *This is an example.*

8. Citations and Bibliography

The `jmlr` class automatically loads `natbib`. This sample file has the citations defined in the accompanying BibTeX file `pmlr-sample.bib`. For a parenthetical citation use `\citep`. For example (?). For a textual citation use `\citet`. For example ?. Both commands may take a comma-separated list, for example ??.

These commands have optional arguments and have a starred version. See the `natbib` documentation for further details.⁴

The bibliography is displayed using `\bibliography`.

Acknowledgments

Acknowledgements go here.

References

Appendix A. First Appendix

This is the first appendix.

Appendix B. Second Appendix

This is the second appendix.

4. Either `texdoc natbib` or <http://www.ctan.org/pkg/natbib>