Mobile Robots Pose Tracking

A Set-membership Approach using a Visibility Information

Rémy Guyonneau, Sébastien Lagrange and Laurent Hardouin
Laboratoire d’Ingénierie des Systèmes Automatisés (LISA), Université d’Angers, 62 avenue Notre Dame du Lac, Angers, France

Abstract: This paper proposes a set-membership method based on interval analysis to solve the pose tracking problem. The originality of this approach is to consider weak sensors data: the visibility between two robots. By using a team of robots and this boolean information (two robots see each other or not), the objective is to compensate the odometry errors and be able to localize, in a guaranteed way, the robots in an indoor environment. This environment is supposed to be defined by two sets, an inner and an outer characterizations. Simulated results allow to evaluate the efficiency and the limits of the proposed method.

1 INTRODUCTION

Robot localization is an important issue in mobile robotics (J. Borenstein, 1996; M.J. Segura, 2009; J. Zhou, 2011) since it is one of the most basic requirement for many autonomous tasks. The objective is to estimate the pose (position and orientation, Figure 1) of a mobile robot by using the knowledge of an environment (e.g. a map) and sensors data.

In this paper the pose tracking problem is considered: the objective is to compute the current pose of a robot knowing its previous one and avoiding its drifting. To compensate the drifting, due to odometry\(^1\) errors, external data are necessary. Contrary to most of the localisation approaches that use range sensors (P. Jensfelt, 2001; K. Lingemann, 2005; Abeles, 2011) this paper tends to prove that weak informations can lead to an efficient localization too. The chosen\(^1\) information is the visibility between two robots: two robots are visible if there is no obstacle between them, else there are not visible. This is a boolean information defined in Section 2.1.

Note that this information does not depend of the robots’ orientation. That is the reason why \(\theta_i\) is assumed to be given by a compass. The objective is then to estimate the position \(\mathbf{x}_i = (x_i, x_2)^T\) of a robot \(r_i\).

A robot \(r_i\) is characterized by the following discrete time dynamic equation:

\[
\mathbf{q}_i(k+1) = f(\mathbf{q}_i(k), \mathbf{u}_i(k))
\]

(1)

with \(k\) the discrete time, \(\mathbf{q}_i(k) = (x_i(k), \theta_i(k))\) the pose of the robot, \(\mathbf{x}_i(k) = (x_1(k), x_2(k))^T\) its position, and \(\mathbf{u}_i(k)\) the input vector (associated with the odometry and the compass). The function \(f\) characterizes the robot’s dynamics. In order to exploit the visibility information a team of \(m\) robots \(\mathbb{R}_t = \{r_1, \cdots, r_i, \cdots, r_m\}\) have to be considered.

In this paper the environment is assumed to be an indoor environment \(E\) composed by \(n\) obstacles \(E_j\), \(j = 1, \cdots, n\). This environment is not known but is characterized by two known sets (see Figure 2): \(E^-\), an inner characterization, and \(E^+\) an outer characterization, such as \(E^- \subset E \subset E^+\).

To solve this problem a set-membership approach of the localization problem based on interval analysis is considered (E. Seignez, 2005; Jaulin, 2009). In this context the the LUVIA algorithm (Localisation Updating with Visibility and Interval Analysis) has been developed to solve the pose tracking problem.

---

\(^{1}\)Sensors that compute the moves of a robot.
2 ALGEBRAIC TOOLS

This section introduces some algebraic tools needful in this paper. First the visibility is defined in Section 2.1. Then interval analysis is presented in Section 2.2 and the environment characterizations are presented in Section 2.3.

2.1 The Visibility

The considered weak information to solve the pose tracking problem is the visibility between two robots. This corresponds to two binary relations: the visibility relation and the non-visibility relation (Definitions 1 and 4). Figure 3 shows an example of visibility informations.

Remark 3. \( \nabla \) is not transitive since \((x_1 \nabla x_2) \nabla (x_2 \nabla x_3) \Rightarrow (x_1 \nabla x_3) \). 

Definition 1. The visibility between two robots \( r_1 \) and \( r_2 \) with their respective positions \( x_1 \) and \( x_2 \) in an environment \( \mathcal{E} \) is a binary relation noted \( \nabla \) such as

\[
(x_1 \nabla x_2)_{\mathcal{E}} \iff \text{Seg}(x_1, x_2) \cap \mathcal{E} = \emptyset.
\] (2)

Properties 2.
- \( \nabla \) is symmetric, i.e., \( \forall x_1, x_2, (x_1 \nabla x_2)_{\mathcal{E}} \Rightarrow (x_2 \nabla x_1)_{\mathcal{E}} \).
- \( \nabla \) is reflexive, i.e., \( \forall x, (x \nabla x)_{\mathcal{E}} \).

Remark 4. \( \nabla \) is not transitive since \((x_1 \nabla x_2)_{\mathcal{E}} \) and \((x_2 \nabla x_3)_{\mathcal{E}} \Rightarrow (x_1 \nabla x_3)_{\mathcal{E}} \).

Definition 4. The non-visibility between two robots \( r_1 \) and \( r_2 \) with their respective positions \( x_1 \) and \( x_2 \) in an environment \( \mathcal{E} \) is a binary relation noted \( \overline{\nabla} \) such as

\[
(x_1 \overline{\nabla} x_2)_{\mathcal{E}} \iff \text{Seg}(x_1, x_2) \cap \mathcal{E} \neq \emptyset.
\] (3)

Properties 5.
- \( \overline{\nabla} \) is the complement of \( \nabla \).
- \( \overline{\nabla} \) is symmetric, i.e., \( \forall x_1, x_2, (x_1 \overline{\nabla} x_2)_{\mathcal{E}} \Rightarrow (x_2 \overline{\nabla} x_1)_{\mathcal{E}} \).

2.2 Interval Analysis

An interval is a closed subset of \( \mathbb{R} \), noted \([a, b] = [a, b] \), with \( a \) its lower bound and \( b \) its upper bound. An interval vector (L. Jaulin, 2001; R. E. Moore, 2009), or a box, \([x]\) is defined as a closed subset of \( \mathbb{R}^n \): \( \forall x \in [x] = \{(x_1, x_2, \ldots, x_n) \mid [x_1], [x_2], \ldots, [x_n] \} \).

The size of a box is defined as

\[
\text{Size}([x]) = (x_1 - x_1) \times (x_2 - x_2) \times \cdots. \tag{4}
\]

It can be noticed that any arithmetic operators such as +, −, ×, ÷ and functions such as exp, sin, sqrt, ... can be easily extended to intervals (Neumaier, 1991). The hull of a set of intervals corresponds to the smallest connected interval that enclosed all the intervals. It can be easily extended to interval vectors. The Bisect() function divides an interval \([x]\) into two intervals \([x_1]\) and \([x_2]\) such as \([x_1] \cup [x_2] = [x]\), \([x_1] \cap [x_2] = \emptyset\) and \(\text{Size}([x_1]) = \text{Size}([x_2])\). As for the hull, the bisection can be extended to interval vectors.

2.3 The Environment Characterization

As said in Section 1, the environment \( \mathcal{E} \) is characterized by two sets \( \mathcal{E}^{-} \) and \( \mathcal{E}^{+} \). In this paper those sets are assumed to be interval segment sets. In this section interval segments are first defined then a final definition of \( \mathcal{E}^{-} \) and \( \mathcal{E}^{+} \) is given.

Definition 7. Let \([x_1]\) and \([x_2]\) be two boxes, an interval segment (see Figure 4) is defined by

\[
\text{Seg}([x_1], [x_2]) = \{\text{Seg}(x_1, x_2) \mid x_1 \in [x_1], x_2 \in [x_2]\}
\] (5)

Note that it is possible to extend the segment intersection to interval segments.
Definition 9. Let $[x_1]$ and $[x_2]$ be two boxes, and an environment $E$

$$([x_1] \cup [x_2])_E \implies \forall x_1 \in [x_1], \forall x_2 \in [x_2], (x_1 \cup x_2)_E,$$ (10)

$$([x_1] \cap [x_2])_E \implies \forall x_1 \in [x_1], \forall x_2 \in [x_2], (x_1 \cap x_2)_E.$$ (11)

Remark 10. Note that $(([x_1] \cup [x_2])_E)$ and $(([x_1] \cap [x_2])_E)$ can be both false as depicted in Figure 6.

Figure 6: In this example $(([x_1] \cup [x_2])_E)$ and $(([x_1] \cap [x_2])_E)$ are both false (since $([x_1] \cup [x_2])_E$) and $(([x_1] \cap [x_2])_E$).

Lemma 11. Let $r_1$ and $r_2$ be two robots with their respective positions $x_1 \in [x_1]$ and $x_2 \in [x_2]$ and an environment $E$ with an inner approximation $E^-$:

$$(x_1 \cup x_2)_E \implies ([x_1] \cup [x_2])_E^- - is false. \quad (12)$$

Proof. $$(x_1 \cup x_2)_E \Rightarrow Seg(x_1, x_2) \cap E = \emptyset,$$

$$\Rightarrow Seg(x_1, x_2) \cap E^- = \emptyset \text{ since } E^- \subseteq E,$$

$$\Rightarrow (x_1 \cup x_2)_E^- = \emptyset,$$

$$\Rightarrow \exists i \in [x_1], \exists x_2 \in [x_2] \mid (x_i \cup x_2)_E^-,$$

$$\Rightarrow ([x_1] \cup [x_2])_E^- - is false.$$ \hfill \square

Lemma 12. Let $r_1$ and $r_2$ be two robots with their respective positions $x_1 \in [x_1]$ and $x_2 \in [x_2]$ and an environment $E$ with an outer approximation $E^+$:

$$(x_1 \cap x_2)_E \Rightarrow ([x_1] \cap [x_2])_E^+ - is false. \quad (13)$$

Proof. $$(x_1 \cap x_2)_E \Rightarrow Seg(x_1, x_2) \cap E \neq \emptyset,$$

$$\Rightarrow Seg(x_1, x_2) \cap E^+ \neq \emptyset \text{ since } E \subseteq E^+,$$

$$\Rightarrow (x_1 \cap x_2)_E^+ = \emptyset,$$

$$\Rightarrow \exists x_1 \in [x_1], \exists x_2 \in [x_2] \mid (x_1 \cap x_2)_E^+, \Rightarrow ([x_1] \cap [x_2])_E^+ - is false.$$ \hfill \square

Lemma 13. Let $r_1$ and $r_2$ be two robots with their respective positions $x_1 \in [x_1]$ and $x_2 \in [x_2]$ and an environment $E$ with an outer approximation $E^+$:

$$(x_1 \cup x_2)_E^+ \implies \forall j, \text{Intersect}(Seg([x_1], [x_2]), E^+) > 0. \quad (14)$$

Proof. $$(x_1 \cup x_2)_E^+ \Rightarrow \forall x_1 \in [x_1], \forall x_2 \in [x_2], (x_1 \cup x_2)_E^+,$$

$$\Rightarrow \forall x_1 \in [x_1], \forall x_2 \in [x_2], \forall j, \text{Intersect}(Seg(x_1, x_2), E^+) > 0,$$

$$\Rightarrow \forall j, \text{Intersect}(Seg([x_1], [x_2]), E^+) > 0.$$ \hfill \square
Lemma 14. Let \( r_1 \) and \( r_2 \) be two robots with their respective positions \( x_1 \in [x_1] \) and \( x_2 \in [x_2] \) and an environment \( E \) with an inner approximation \( E^- \):
\[
(x_1 \bigvee V(x_2))_E^- \Rightarrow \exists j \mid \text{Intersect}(\text{Seg}([x_1], [x_2]), E_j) < 0
\]  
(15)

Proof. \( (x_1 \bigvee V(x_2))_E^- \Rightarrow \forall x_1 \in [x_1], x_2 \in [x_2], (x_1 \bigvee V(x_2))_E^- \).
\[ \Rightarrow \exists j \mid \forall x_1 \in [x_1], \forall x_2 \in [x_2], \text{Intersect}(\text{Seg}([x_1], [x_2]), E_j) < 0, \]
\[ \Rightarrow \exists j \mid \text{Intersect}(\text{Seg}([x_1], [x_2]), E_j) < 0. \]

From those lemmas it is possible to deduce the following propositions:

Proposition 15. Let \( r_1 \) and \( r_2 \) be two robots with their respective positions \( x_1 \in [x_1] \) and \( x_2 \in [x_2] \) and an environment \( E \) with an inner approximation \( E^- \):
\[
(x_1 \bigvee V(x_2))_E \Rightarrow \forall j, \text{Intersect}(\text{Seg}([x_1], [x_2]), E_j) > 0
\]  
(16)

Proof. \( (x_1 \bigvee V(x_2))_E \Rightarrow \exists j \mid \text{Intersect}(\text{Seg}([x_1], [x_2]), E_j) > 0 \).

Proposition 16. Let \( r_1 \) and \( r_2 \) be two robots with their respective positions \( x_1 \in [x_1] \) and \( x_2 \in [x_2] \) and an environment \( E \) with an outer approximation \( E^+ \):
\[
(x_1 \bigwedge V(x_2))_E \Rightarrow \exists j \mid \text{Intersect}(\text{Seg}([x_1], [x_2]), E_j^+) \leq 0
\]  
(17)

Proof. \( (x_1 \bigwedge V(x_2))_E \Rightarrow (x_1 \bigwedge V(x_2))_E^+ \) is false (Lemma 12), \( \Rightarrow (\forall j, \text{Intersect}(\text{Seg}([x_1], [x_2]), E_j^+) > 0) \) is false (Lemma 13), \( \Rightarrow \exists j \mid \text{Intersect}(\text{Seg}([x_1], [x_2]), E_j^+) \leq 0. \)

Each robot \( r_i \in \mathcal{R} \) does a measurement vector \( y_i(k) \) at time \( k \).
\[
y_i(k) = (y_{i1}(k), \ldots, y_{ir_i}(k), \ldots, y_{im}(k))^T
\]  
(18)

with \( y_{ir_i}(k) \in \{\text{true, false}\} \) the visibility between the robots \( r_i \) and \( r_j \) at time \( k \). Note that \( y_{ir_i}(k) = \text{true} \) means \( (x_i \bigvee V(x_j))_E \) while \( y_{ir_i}(k) = \text{false} \) means \( (x_i \bigvee V(x_j))_E \).

Algorithm 1 computes the robots’ pose in a bounded error context and for each robot the LUVIA algorithm (Algorithm 2) contracts the robot’s estimated pose to all consistent values according to the environment approximations and the visibility measurements. Note that Algorithm 3 performs a visibility test using the Propositions 15 and 16. This algorithm has three possible return values:
- \( \text{true} \) if \( (x_i \bigvee V(x_j))_E^+ \),
- \( \text{false} \) if \( (x_i \bigvee V(x_j))_E^- \),
- \( \text{indeterminate} \) if no conclusion can be done.

Algorithm 1: The pose tracking algorithm.

<table>
<thead>
<tr>
<th>Data</th>
<th>( \mathcal{R}, E^-, E^+ )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r_i \in \mathcal{R} )</td>
<td>initialize ( q_i(0) )</td>
</tr>
<tr>
<td>for ( k = 1 ) to end do</td>
<td></td>
</tr>
<tr>
<td>( r_i \in \mathcal{R} )</td>
<td>( y_i(k) = f_i([q_i(k-1)], [u_i(k-1)]) )</td>
</tr>
<tr>
<td>( r_i \in \mathcal{R} )</td>
<td>( y_i(k) ) = new measurement set;</td>
</tr>
<tr>
<td>( r_i \in \mathcal{R} )</td>
<td>LUVIA((r_i, y_i(k), \mathcal{R}, E^-, E^+); )</td>
</tr>
<tr>
<td>end</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>( \mathcal{R}_i )</td>
</tr>
</tbody>
</table>

5 RESULTS AND CONCLUSIONS

In order to test this pose tracking approach, a simulator has been developed. The simulated environment has a 10 \( \times \) 10m size, see Figure 7. At each iteration a robot does a 10cm distance move with a bounded error of \( \pm 0.1 \text{cm} \) and a bounded compass error of \( \pm 2.5 \text{deg} \). In the later, the results are obtained for 1500 iterations of the pose tracking algorithm. Note that \( \forall r_i \in \mathcal{R}, \text{Size}([x_i(0)]) = 1 \text{m}^2 \) and \( x_i(0) \in [x_i(0)] \), with \( x_i(0) \) the initial position of \( r_i \).

The processor used for the simulations has the following characteristics:

Intel(R) Core(TM)2 CPU - 6420 @ 2.13GHz.

5.1 Influence of the Number of Robots

The objective it to evaluate the effect of the number of robots for the localization. The considered environment is \( E = \bigcup_{j=1}^6 E_j \). Figure 8 represents the obtained results.
Algorithm 2: LUVIA algorithm.

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Data: ( r, y, x, \hat{e}^-, \hat{e}^+ )</td>
</tr>
<tr>
<td>2</td>
<td>( L = \emptyset ), ( L_{OK} = \emptyset )</td>
</tr>
<tr>
<td>3</td>
<td>while ( L.\text{Size}(i) &gt; 0 ) do</td>
</tr>
<tr>
<td>4</td>
<td>( x_i = L.\text{Pop}_\text{out}() );</td>
</tr>
<tr>
<td>5</td>
<td>( \text{bisect} = \text{false} );</td>
</tr>
<tr>
<td>6</td>
<td>for ( r_j \in \mathcal{R} ) (with ( i \neq j )) do</td>
</tr>
<tr>
<td>7</td>
<td>( \text{if} \ y_j \neq \text{true} ) then</td>
</tr>
<tr>
<td>8</td>
<td>( \text{consistent} = )</td>
</tr>
<tr>
<td>9</td>
<td>( \text{Visibility}_\text{Test}([x], [x_j(k)], \hat{e}^-) );</td>
</tr>
<tr>
<td>10</td>
<td>else</td>
</tr>
<tr>
<td>11</td>
<td>( \text{consistent} = )</td>
</tr>
<tr>
<td>12</td>
<td>( \text{Visibility}_\text{Test}([x], [x_j(k)], \hat{e}^+) );</td>
</tr>
<tr>
<td>13</td>
<td>( \text{if} \ \text{consistent} = \text{true} ) then</td>
</tr>
<tr>
<td>14</td>
<td>( \text{consistent} = \text{false} );</td>
</tr>
<tr>
<td>15</td>
<td>else if ( \text{consistent} = \text{false} ) then</td>
</tr>
<tr>
<td>16</td>
<td>( \text{consistent} = \text{true} );</td>
</tr>
<tr>
<td>17</td>
<td>end</td>
</tr>
<tr>
<td>18</td>
<td>end</td>
</tr>
<tr>
<td>19</td>
<td>( \text{if} \ \text{consistent} = \text{false} ) then</td>
</tr>
<tr>
<td>20</td>
<td>( \text{break} );</td>
</tr>
<tr>
<td>21</td>
<td>else if ( \text{consistent} = \text{indeterminate} ) then</td>
</tr>
<tr>
<td>22</td>
<td>( \text{bisect} = \text{true} );</td>
</tr>
<tr>
<td>23</td>
<td>end</td>
</tr>
<tr>
<td>24</td>
<td>if ( \text{consistent} \neq \text{false} ) then</td>
</tr>
<tr>
<td>25</td>
<td>( \text{if} \ \text{bisect} = \text{true} ) and ( \text{Size}([x]) &gt; \varepsilon ) then</td>
</tr>
<tr>
<td>26</td>
<td>( \text{Bisect}([x], L) );</td>
</tr>
<tr>
<td>27</td>
<td>end</td>
</tr>
<tr>
<td>28</td>
<td>else</td>
</tr>
<tr>
<td>29</td>
<td>( L_{OK}.\text{Push}_\text{back}([x]) );</td>
</tr>
<tr>
<td>30</td>
<td>end</td>
</tr>
<tr>
<td>31</td>
<td>end</td>
</tr>
<tr>
<td>32</td>
<td>end</td>
</tr>
<tr>
<td>33</td>
<td>Result: ( \text{Hull}(L_{OK}) ).</td>
</tr>
</tbody>
</table>

Algorithm 3: Visibility\_test.

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( \text{visible} = \text{true} );</td>
</tr>
<tr>
<td>2</td>
<td>for ( \hat{e}_k^+ \in \hat{e}^+ ) do</td>
</tr>
<tr>
<td>3</td>
<td>( \text{inter} = \text{Intersect}(\text{Seg}([x_1], [x_2]), \hat{e}_k^+) );</td>
</tr>
<tr>
<td>4</td>
<td>( \text{if} \ \text{inter} &lt; 0 ) then</td>
</tr>
<tr>
<td>5</td>
<td>( \text{visible} = \text{false} );</td>
</tr>
<tr>
<td>6</td>
<td>( \text{break} );</td>
</tr>
<tr>
<td>7</td>
<td>else if ( \text{inter} \neq 0 ) then</td>
</tr>
<tr>
<td>8</td>
<td>( \text{visible} = \text{indeterminate} );</td>
</tr>
<tr>
<td>9</td>
<td>end</td>
</tr>
<tr>
<td>10</td>
<td>Result: ( \text{visible} ).</td>
</tr>
</tbody>
</table>

It appears that for a given environment a minimal number of robots is necessary to perform an efficient pose tracking. It can be explained by the fact that with few robots, \( y_i(k) \) carries few information. In this configuration, at least 7 robots are necessary to perform an efficient localization.

On the other hand, too many robots do not improve significantly the localization but increase the computation time. This localization maximal precision is directly dependent of \( \hat{e}^- \) and \( \hat{e}^+ \). In this example, over 9 robots the results are similar.

![Figure 8: Results over 1500 iterations.](image)

5.2 Influence of the Number of Obstacles

In this section the influence of the number of obstacles over the localization results is evaluated. A team of 7 robots is considered \( \mathcal{R} = \{ r_i \}, i = 1, \ldots, 7 \). Figure 9 represents the obtained results.

It appears that for a given number of robots it exists a minimal and a maximal number of obstacles that allow to perform an efficient localization. It can be explained by the fact that without any obstacle, the robots see each other all the time, so the visibility sensor returns always the same value and does not provide useful information. It is the same argument with too many obstacles.

In Figure 9 it is possible to see that under 4 ob-
stcles and over 8 obstacles, the pose tracking does not lead to an efficient localization of the robots. It also appears that 7 obstacles do not lead to an efficient pose tracking. Hence the success of the pose tracking depends on the positions and the sizes of the obstacles in the environment.

5.3 Conclusions

In this paper it is shown that using interval analysis it is possible to perform a pose tracking of mobile robots even assuming weak informations as the visibility between robots. The LUVIA algorithm is a guaranteed algorithm that exploits this boolean information.

It appears in Section 5.2 that characterizing the environments by counting the number of obstacles is not pertinent here. In a future work it could be interesting to characterize the environment by visibility zones allowing to calculate a minimal number of robots required to perform a pose tracking, according to the number and/or the size of the zones.

Finally it could be interesting to process an experimentation with actual robots.

REFERENCES


APPENDIX

Segment Intersection

The function $\text{Intersect}()$. Definition 17, allows to test the intersection between two segments.

**Definition 17.** Let $\text{Seg}(x_1, x_2)$ and $\text{Seg}(x_3, x_4)$ be two segments, the function

$$\text{Intersect}(\text{Seg}(x_1, x_2), \text{Seg}(x_3, x_4))$$

is defined by

$$\text{Intersect}(\text{Seg}(x_1, x_2), \text{Seg}(x_3, x_4)) = \text{Max}(\text{Side}(x_1, \text{Seg}(x_3, x_4)) \cdot \text{Side}(x_2, \text{Seg}(x_3, x_4)), \text{Side}(x_3, \text{Seg}(x_1, x_2)) \cdot \text{Side}(x_4, \text{Seg}(x_1, x_2))).$$

where $\text{Side}()$. Definition 18, allows to test the side of a point with a segment.

**Definition 18.** Let $\text{Seg}(x_1, x_2)$ be a segment and $x_3$ be a point, the function $\text{Side}(x_3, \text{Seg}(x_1, x_2))$ is defined by

$$\text{Side}(x_3, \text{Seg}(x_1, x_2)) = \text{det}(x_1 - x_3, x_2 - x_1),$$

with $\text{det}$ the determinant.

Figure 9 represents three intersection tests.