Self-localization of Anonymous Mobile Robots from Aerial Images
Olivier Poulet, François Guérin, Frédéric Guinand

To cite this version:

HAL Id: hal-02122623
https://hal.archives-ouvertes.fr/hal-02122623
Submitted on 7 May 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Self-localization of Anonymous Mobile Robots from Aerial Images

Olivier Poulet and François Guérin and Frédéric Guinand

Abstract—This paper presents three methods for anonymous mobile robots localization within a global frame. An aerial camera takes, at regular time intervals, pictures of the area in which robots are moving. The camera determines the coordinates of each robot. Each robot receives the whole set of coordinates extracted from each picture. Mobile Robots are all identical, they do not have any identifier and they can neither communicate with each other nor they can detect themselves. The first localization method is based on the analysis of the angular variation between two images. The second method relies on the analysis of the distances stemmed from three successive pictures. The last one determines if there exists an orientation allowing a specific robot to travel the path between two successive positions. A simulation platform using augmented reality and the multi-robot software Player-Stage are presented. This platform is used for validating the different localization methods. Tests and results are presented and compared.

I. INTRODUCTION

Within the mobile robotics domain, the generic term of localization may have different meanings according to the context.

Individual localization concerns the ability of a robot to localize itself alone. Several methods are able to perform individual localization. The dead reckoning method [1] allows a robot, knowing its initial position and its own movements, to compute, at any moment, its new relative position. However, the lack of accuracy limits the applicability of this method. Another method, called global localization and described in [2], relies on landmarks distributed in the environment of the robot. Thanks to these landmarks and without any knowledge about its previous positions, the robot is able to localize itself. This method uses lidars, 3D camera and allows to do SLAM. The kidnapping method is the last individual localization method. It allows a robot to re-localize itself after being moved from a known position to an unknown one [3].

In a multi-robots setting, the localization problem becomes more tricky since each robot has to identify itself on the picture. This identification is not so easy because different robots in the scene may performed close and symmetric movements. This process is however necessary for many applications like exploration [4] or transportation [5].

The methods of cooperative localization in a global frame let robots adjust their own position from the knowledge of relative positions and directions of the other robots of the group. In [6], the authors use a Monte-Carlo method that provides an accuracy much better than global localization methods for one robot. In addition the method is much faster. Mutual Localizations (ML) are used for localizing neighbor robots with respect to fixed or mobile landmarks attached to the robots [7].

In the literature, some works consider anonymous robots. In that case, robots are identical, they cannot be distinguished by their shape or any other characteristics. This leads to additional complexity for solving the localization problem. The number of existing works addressing the ML problem, in this context, is quite limited and the researches of Franchi et al. [7] are, to the best of our knowledge, among the most advanced ones focusing on this specific topic. Each robot considered in their work is equipped with some material allowing this robot to self-estimate its pose (position and orientation) in a fixed frame associated to it. In addition, each robot has a detector module that allows it to detect the other robots that are within its neighborhood. Finally, each robot is able to communicate with other robots that are within their neighborhood. The authors propose in their work a two-stages method. In the first phase, each robot sends to its neighbors its index, the estimated pose for itself and the relative pose measures of its neighbors (without associating the measures with index) produced by its detector, and receives similar information from its neighbors. These information are used for computing all possible relative positions of the robots. Then, in the second phase the best possible solution is estimated thanks to an extended Kalman filter. The algorithm takes into account false positive, i.e. objects looking like robots but that are not robots, and false negative corresponding to non detected robots because of obstacles that may hide them. Anonymity entails some ambiguous solutions for estimating the pose of other robots, especially when their configuration has some symmetric properties. For addressing these issues, Franchi et al. proposed a new method by substituting the Kalman filter by a particle filter [8].

Cognetti et al. have extended the results for the problem of anonymous localization in three dimensions in [9]. Finally in [10], the authors describe algorithms for anonymous localization for different scenarios: in two and three dimensions with linear and angular speeds measures, and in three dimensions with speed-up and angular speed measures. The initial algorithms, based on geometric computations, gather the data of each robot and data received from other robots for providing different possible figures.

In the previously mentioned works, communications between robots are anonymous but use indices. The use of
indices makes it possible to avoid an over-representation of some data. If the same set of data is taken into account many times, the process might converge towards a wrong position as it was outlined in [11].

The main problem addressed in the current work is the global localization of anonymous and identical robots. It is assumed that some pictures are taken, from a camera, at regular time intervals and that the coordinates, in the camera frame, of all the robots can be extracted from the pictures. Another hypothesis is that all the robots are on each picture. The set of anonymous coordinates is sent to all the robots. Using these anonymous sets illustrated in Figure 1, each robot has to determine its own position, without communicating with the other robots. The localization is made in a global frame, the one of the camera. Considering our set of assumptions, in particular the anonymity, the current work is, to the best of our knowledge, among the first one addressing this problem.

In Section II, three localization methods addressing this problem are described. The first method compares angular variations of robots between consecutive pictures, while linear speed variations are at the center of the second method. The last method looks for possible orientations for the robots based on the set of coordinates received from each picture. Experiments and results will be presented in Section III and will be followed by the conclusion.

![Fig. 1. The two robots receive, at regular interval, the coordinates of all the robots on the picture. Robots have to determine who they are.](image1)

**II. LOCALIZATION METHODS**

**A. Angular variation**

This method relies on the angles of robots with respect to the frame of the camera. For its localization, each robot records all the angles extracted from the pictures provided by the camera. Between the reception of two consecutive pictures, each robot performs a rotation. Upon reception of the new picture it computes the new angles for comparing them with the previous ones. The difference between measured angles on the pictures and the estimated rotation angles gives some clues for answering the question: which robot am i on the picture?

The angles sending by the camera are $\theta_i(p)$ for the picture $i$ and the robot detected $p$. To determine its self-localization, a robot needs to compare, if we assume the linear speed of robots equal to zero between two pictures, for $k = 1$ to $n$ ($n$ is the number of angles detected by the camera and send to each robot):

$$S_k = |r\Delta t - (\theta_j(k) - \theta_i(k))|$$

Where $i$ and $j$ are two consecutive pictures, $\Delta t$ is the time between two pictures, $r$ is the rotation speed of the robot ($r\Delta t < 2\pi$). The smallest $S_k$ define the position $k$ of the robot on the pictures.

**B. Linear speed variation**

In the pinhole model [13], the camera has its own Cartesian coordinate system in three dimensions ($X, Y, Z$). The origin moves on a plane and the projection of the image plan is denoted by $r$ as illustrated on Figure 2.

![Fig. 2. The considered pinhole model.](image2)

The coordinates of point $R$ in the camera frame are:

$$R = \begin{bmatrix} X_r \\ Y_r \\ Z_r \end{bmatrix}$$

(2)

The coordinates of the projection of the robot on image plan are:

$$r = \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = \lambda \begin{bmatrix} X_r \\ Y_r \\ Z_r \end{bmatrix}$$

(3)

with $\lambda = \frac{f}{z_r}$ where $f$ is the focal length of the camera.

If the robot moves from $R$ to another $R'$ position ($R' = (X'_r, Y'_r, Z'_r)$), then:

$$r' = \begin{bmatrix} x'_r \\ y'_r \\ z'_r \end{bmatrix} = \lambda' \begin{bmatrix} X'_r \\ Y'_r \\ Z'_r \end{bmatrix}$$

(4)

with $\lambda' = \frac{f'}{z'_r}$. The distance from $R$ to $R'$ is:

$$d_0 = \sqrt{(X_r - X'_r)^2 + (Y_r - Y'_r)^2 + (Z_r - Z'_r)^2}$$

(5)

and on the image plan we have:

$$d'_0 = \sqrt{(\lambda X_r - \lambda' X'_r)^2 + (\lambda Y_r - \lambda' Y'_r)^2 + (\lambda Z_r - \lambda' Z'_r)^2}$$

(6)

If both plans (image plan and robots plan) are parallel, then $d'_0 = \lambda \sqrt{(X_r - X'_r)^2 + (Y_r - Y'_r)^2}$ and thus,
can computes the value of $a$ again until it receives a third picture. By recording its speeds second picture, it modifies its speed and moves linearly between three pictures, correspond-

$$d'_0 = \sqrt{(x_r - x'_r)^2 + (y_r - y'_r)^2}$$  \hspace{1cm} (7)$$

The moving distances between three pictures, corresponding to dates $t_0$, $t_1$ and $t_2$, can be computed according to the Thalès theorem (as illustrated on Figure 3), under the assumption that speeds are linear and constant between two consecutive dates ($v_0$ between $t_0$ and $t_1$ and $v_1$ for the interval $[t_1, t_2]$):

$$a = \frac{d_0}{d_1} = \frac{d'_0}{d'_1} = \frac{v_0}{v_1}$$  \hspace{1cm} (8)$$

Each time the robot moves a distance $d$, the camera will measure the distance $a \times d$ on the image plan of the camera.

For its own localization, a robot, when receiving a picture, moves linearly at a given speed. Upon reception of the second picture, it modifies its speed and moves linearly again until it receives a third picture. By recording its speeds between pictures 1 and 2 and between pictures 2 and 3, it can compute the value of $a$. Finally, by comparing the set of ratios stemmed from all the coordinates for all the robots sent by the camera, it can deduce where is its own position on each picture. The computational complexity depends on the number of robots, for $n$ robots, this complexity, in number of tests, is $O(n^3)$.

C. Localization during a random motion

The previous methods are based on the reception of pictures from a camera that thus induces a synchronization of the process. In this third method, each robot works independently of the rhythm at which the pictures are received. Robots are characterized by a linear and an angular speeds ($v, r$). The well known unicycle-like model gives for the robot plan:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & 0 & 0 \\ \sin(\theta) & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ r \end{pmatrix}$$  \hspace{1cm} (9)$$

and for the image plan of the camera:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & 0 & 0 \\ \sin(\theta) & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ r \end{pmatrix}$$  \hspace{1cm} (10)$$

The robot moves along a trajectory as illustrated on Figure 4. The first picture is taken at time $m$ and the position is denoted by $p_m$. The second picture is taken at time $n$ at position $p_n$. Both the image plan of the camera and the robots plan are supposed to be parallel.

If $f$ is a function of classes $C^n$ over $I$, then, from the Taylor theorem:

$$f(x_0 - h) = f(x_0) - hf'(x_0) + he(h) \quad \forall h \in \mathbb{R}$$

with $x_0 \in I$ and such that $\lim_{h \to 0} e(h) = 0$.

By setting $x = t$ and $h = \delta t$, and $x(t_j) = x_j$, $x(t_j - \delta t) = x_{j-1}$, $y(t_j) = y_j$, $y(t_j - \delta t) = y_{j-1}$ and $\theta(t_j) = \theta_j$, $\theta(t_j - \delta t) = \theta_{j-1}$, we obtain at point $n$:

$$\begin{pmatrix} x_n \\ y_n \\ \theta_n \end{pmatrix} = \begin{pmatrix} x_{n-1} \\ y_{n-1} \\ \theta_{n-1} \end{pmatrix} + \delta t \begin{pmatrix} \lambda \cos(\theta_{n-1}) & 0 & 0 \\ \lambda \sin(\theta_{n-1}) & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_{n-1} \\ r_{n-1} \end{pmatrix}$$  \hspace{1cm} (11)$$

By applying the same last formula at $x_{n-1}$, we obtain:

$$\begin{pmatrix} x_{n-1} \\ y_{n-1} \\ \theta_{n-1} \end{pmatrix} = \begin{pmatrix} x_{n-2} \\ y_{n-2} \\ \theta_{n-2} \end{pmatrix} + \delta t \begin{pmatrix} \lambda \cos(\theta_{n-2}) & 0 & 0 \\ \lambda \sin(\theta_{n-2}) & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_{n-2} \\ r_{n-2} \end{pmatrix}$$  \hspace{1cm} (12)$$

By iteration, this leads to:

$$\begin{pmatrix} x_n \\ y_n \\ \theta_n \end{pmatrix} = \begin{pmatrix} x_m \\ y_m \\ \theta_m \end{pmatrix} + \delta t \sum_{j=m+1}^{n} \begin{pmatrix} \lambda \cos(\theta_j) & 0 & 0 \\ \lambda \sin(\theta_j) & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_j \\ r_j \end{pmatrix}$$  \hspace{1cm} (13)$$

Then, focusing only on the last line:
\[ \theta_n = \theta_m + \delta t \sum_{j=m+1}^{n} r_j \quad \text{or} \quad \theta_j = \theta_m + \delta t \sum_{k=m+1}^{j} r_k \]  

(14)

and by reinjecting this into equation (13), we obtain:

\[
\begin{pmatrix}
x_n \\
y_n
\end{pmatrix} = \begin{pmatrix} x_m \\ y_m \end{pmatrix} + \delta t \sum_{j=m+1}^{n} \left( \lambda \cos(\theta_m + \delta t \sum_{k=m+1}^{j} r_k) \right) v_j
\]

(15)

If \( \lambda \) is known, only \( \theta_m \), the initial angle of the robot on the first picture is unknown. \( r_j \) and \( v_j \) are the rotation and the linear speed commands, they are known by each robot at any moment. Finding the position of a robot on every picture is equivalent to determine if there exist one and only one angle between two sets of coordinates leading this robot from the position \( p_m = (x_m, y_m) \) to \( p_n = (x_n, y_n) \) as illustrated on Figure 5.

The first line of the system (15) is:

\[ x_n = x_m + \delta t \lambda \sum_{j=m+1}^{n} (\cos(\theta_m + \delta t \sum_{k=m+1}^{j} r_k) v_j) \]

(16)

It can be solved in the following way. Let:

\[ A = \sum_{j=m+1}^{n} (\cos(\delta t \sum_{k=m+1}^{j} r_k)) v_j \]

(17)

\[ B = \sum_{j=m+1}^{n} (\sin(\delta t \sum_{k=m+1}^{j} r_k)) v_j \]

(18)

\[ C = \frac{x_n - x_m}{\lambda \delta t} \]

(19)

\[ \rho = \sqrt{A^2 + B^2} \]

\[ \alpha = \arctan \left( \frac{B}{A} \right) \]

(20)

Which leads to:

\[
\begin{align*}
\{ & \text{if } \rho = 0 \text{ there is no solution} \\
& \text{if } | \frac{C}{\rho} | > 1 \text{ there is no solution} \\
& \text{else } \theta_m = \arccos \left( \frac{C}{\rho} \right) - \alpha \\
& \theta_m \in [0; \pi] 
\}
\]

(21)

The second line of the system (15) is:

\[ y_n = y_m + \delta t \lambda \sum_{j=m+1}^{n} (\sin(\theta_m + \delta t \sum_{k=m+1}^{j} r_k) v_j) \]

(22)

That can be solved in the following way. Let \( D = \frac{y_n - y_m}{\lambda \delta t} \),

the equation to be solved is:

\[
\begin{align*}
\{ & \text{if } \rho = 0 \text{ there is no solution} \\
& \text{if } | \frac{D}{\rho} | > 1 \text{ there is no solution} \\
& \text{else } \theta_m = \arcsin \left( \frac{D}{\rho} \right) - \alpha \\
& \theta_m \in \left[ -\frac{\pi}{2}; \frac{\pi}{2} \right] 
\}
\]

(23)

The following cases have to be considered:

\[
\begin{align*}
\{ & \cos(-\theta_m) = \cos(\theta_m) \\
& \sin(\pi - \theta_m) = \sin(\theta_m) 
\}
\]

(24)

When a robot has to localize itself, if \( \lambda \) is known, it compares pairs of positions on the two pictures (one point of the first picture and the second point on the second picture), and attempts to compute a unique initial angle compatible for a motion from \( p_m = (x_m, y_m) \) to \( p_n = (x_n, y_n) \). If such an angle could be determined then the coordinates correspond to the positions of the robot of both images.

If \( \lambda \) is unknown, three pictures are needed. In that case, the angle \( \theta_m \) of the middle picture is common to both trajectories: from \( p_0 \) to \( p_m \) and from \( p_m \) to \( p_n \) as illustrated on Figure 6. In that situation, the system to be solved is:

\[
\begin{align*}
\begin{pmatrix}
x_m \\
y_m
\end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \delta t \sum_{j=1}^{m} \left( \lambda \cos(\theta_m - \delta t \sum_{k=j+1}^{m} r_k) \right) v_j
\end{align*}
\]

(25)

\[
\begin{align*}
\begin{pmatrix}
x_n \\
y_n
\end{pmatrix} = \begin{pmatrix} x_m \\ y_m \end{pmatrix} + \delta t \sum_{j=m+1}^{n} \left( \lambda \cos(\theta_m + \delta t \sum_{k=m+1}^{j} r_k) \right) v_j
\end{align*}
\]
The system has two unknowns: \( \lambda \) and \( \theta_m \). There are \( n^3 \) possibilities for \( n \) robots which entails an increase in the computational complexity.

For the experiments, \( \lambda \) is supposed to be known.

### III. Experiments and Results

Experiments have been led on the multi-robot Player Stage simulator [14]. Robots in the simulator are modeled as simple squares. The simulator runs the movements of the robots that are captured by a webcam taking pictures of the screen of the computer on which the simulation is running, as it is illustrated by a screenshot of the experimental setup on Figure 7.

The coordinates of the robots have been determined using the Artoolkit Software Development Kit for Augmented Reality.

This SDK [2], [12], is able to detect some predefined patterns on a picture and to extract from these patterns a set of coordinates as illustrated on Figure 8.

The Artoolkit software running within Processing 3 detects the robots (basic square patterns) and computes the coordinates of specific points inside these squares (Figure 8). These coordinates are stored in a file which is sent to all the robots. Each of them, upon reception of the file, performs the previously described processing in order to localize itself.

The general principle of the experimental platform is described in Figure 9.

Tests have been realized with two robots. A new picture has been taken every 10 seconds. The camera was positioned approximately at 30cm of the screen. The size of the robots was about 7 mm \( \times \) 7 mm on the screen. About the parallelism of the image plan of the camera and the plan of the robots, a linear constant speed movement has been simulated for a robot between two consecutive pictures. \( \lambda \) was obtained as the ratio between the known distance performed by the robot and the distance measured on the image plan of the camera. This value should have been the same everywhere on the screen, but the parallelism was experimentally not easy to guarantee and was not perfect. Thus, during our experiments, the \( \lambda \) values obtained for different points of the plan of the robots varied between 20.8 and 22.07. An average value was considered for performing the computations. It is worth mentioning that this approximation is a potential source of localization errors.

For the test of angular variation, rotations of robots were randomly generated, started at the same moment, limited to 9 degrees/second and rotating counterclockwise.

For the test of localization by linear speed variation, the robots started at the same time using random linear speeds limited to 0.1 unit/second, one unit being defined within the Player-Stage simulator.

For the test of localization during a movement, the discretization time was \( \delta t = 0.01 \) second. Random trajectories have been programmed on each robot. The linear speed and rotation variations were triggered at random moment (between 0 and 3 seconds each) and bounded by 0.1 and 0.5 unit/second for the linear speed and between -20 and 20 degrees/second for the rotating speed. If a robot met an obstacle (wall, other robot), the movement was stopped and the robot rotated on itself at a random rotating speed.

Results are reported in the table below.

<table>
<thead>
<tr>
<th></th>
<th>angular</th>
<th>linear</th>
<th>movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of runs</td>
<td>65</td>
<td>100</td>
<td>118</td>
</tr>
<tr>
<td>Success rate (%)</td>
<td>96.9</td>
<td>84</td>
<td>82.2</td>
</tr>
<tr>
<td>Average</td>
<td>1.6(^\circ)(^(*))</td>
<td>0.823(^\circ)(^(**))</td>
<td>7.22(^\circ)(^(***))</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>1.1(^\circ)(^(*))</td>
<td>2.93(^\circ)(^(**))</td>
<td>7.3(^\circ)(^(***))</td>
</tr>
</tbody>
</table>

\(^(*)\) average deviation between measured angles and real
ones, (** average deviation between ratio measured and real ones, (***) average angle between x and y axes.

The high success rate for the localization by angular variation could be explained by the limited number of tests that have to be performed for the computation coupled with the efficiency of the Augmented Reality approach. The detection is rather simple and is not error-prone due to the efficiency of the augmented reality method since robots are not moving. However the method relies on the knowledge of the orientation angle, which was made possible by Artoolkit, and this is its main weakness, indeed, if the orientation is not detected the method fails.

The success rate of the method based on the linear speed variation is not as good as for the previous method, however, it remains at a good level. The number of tests needed to be done for the localization is \( n^3 \), where \( n \) is the number of robots.

Finally the last method is more error-prone, however it should be noticed that we made the experiments in the worst conditions by considering similar speeds for the robots that started almost at the same moment. The success rate is however comparable to the results obtained by the previous method for a computational complexity also comparable.

IV. CONCLUSION AND FUTURE WORKS

The aim of this work was to study the problem of the localization of anonymous and identical mobile robots in a multi-robots settings. Robots were not allowed to communicate with each other. Based on the pictures taken from above, for instance a drone taking pictures of several ground robots moving in a specific area, each robot should be able to identify itself in the scene. For that purpose we have proposed three different methods relying on the variation of linear or rotating speeds of the robot. The first method uses the rotation angles of the robots. The rotation angles were extracted from two consecutive pictures sent by the camera. The second method is based on the variation of the linear speed of the robots. This method requires at least three pictures and the synchronization of movements upon reception of pictures. The last method is probably the most robust method since it requires neither a synchronization of the movement nor specific variation in linear and/or rotating speeds. In this last method each robot can behave as it wants and only two images are enough for determining its localization if \( \lambda \) is known.

The methods were tested experimentally through simulation using Player Stage simulator, the Artoolkit SDK for the detection of patterns for extracting the set of coordinates of the robots as well as their orientation angles, and Processing 3 for making everything work. The success rate was always greater than 82%, however this should be confirmed by testing the method with a growing number of robots.

In a short-term perspective, the quality of the localization provided by these methods will be evaluated for real experiments (pictures taken from a drone) and for a large number of robots. Using a drone for localizing ground robots could be used, but it should include a module for automatically correcting the camera tilt variations (gimbal) through an adaptation of the \( \lambda \) value.

In a mid-term perspective, the potential developments for this work are manifold. Localization problems under various situations involving a growing number of robots, or when robots enter into/leave the pictures or when the group behaves like a swarm (similar direction and speeds) are still to be investigated.

REFERENCES