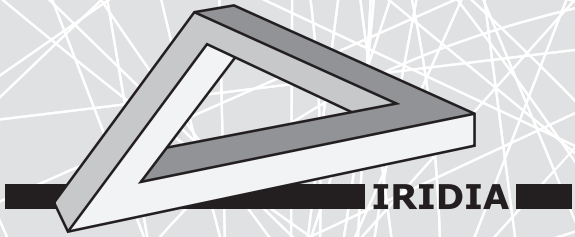


Université Libre de Bruxelles

Institut de Recherches Interdisciplinaires
et de Développements en Intelligence Artificielle

Social Odometry: A Self-Organized Distributed Location Algorithm

Álvaro Gutiérrez, Alexandre Campo, David Fernández, Félix Monasterio-Huelin, Luis Magdalena and Marco Dorigo

IRIDIA – Technical Report Series

Technical Report No.
TR/IRIDIA/2009-014

May 2009

IRIDIA – Technical Report Series
ISSN 1781-3794

Published by:

IRIDIA, *Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle*
Université Libre de Bruxelles
Av F. D. Roosevelt 50, CP 194/6
1050 Bruxelles, Belgium

Technical report number TR/IRIDIA/2009-014

Revision history:

TR/IRIDIA/2009-014.001 May 2009

Social Odometry: A Self-Organized Distributed Location Algorithm

Álvaro Gutiérrez[1], Alexandre Campo[2], David Fernández[1],
Félix Monasterio-Huelin[1], Luis Magdalena[3] and Marco Dorigo[2]

[1] ETSIT, Universidad Politécnica de Madrid, Madrid, Spain
aguti@etsit.upm.es, felix.monasteriohuelin@upm.es, david.fernandez.pastor@alumnos.upm.es
[2] IRIDIA, CoDE, Université Libre de Bruxelles, Bruxelles, Belgium
{acampo,mdorigo}@iridia.ulb.ac.be
[3] European Centre for Soft Computing, Asturias, Spain
luis.magdalena@softcomputing.es

May 2009

Abstract

In this paper, we study a new approach to multi-robot localization based on local communication which confers the robots the possibility to learn from the others. By communicating with the rest of the group, robots are able to correct, localize and achieve tasks they could not solve by their own. We use a foraging task as test bed where a nest and prey areas must be identified by the robots. Once both areas have been located, robots must forage from nest to prey endlessly. Each robot has an estimate of its own location and an associated confidence level that decreases with the distance travelled. The algorithm guides a robot to its goal by imitating estimated locations, confidence levels and actual locations of its neighbors. We evaluate the performance of different local communication algorithms by observing the number of times robots go from nest to prey and come back. We extend the performance analysis to different quantitative measures about the intrinsic recruitment and self-organized process.

1 Introduction

Most of the localization existing works address the problem of a single robot, and only few works address it from the multi-robot point of view. In this way, each single robot could implement its own localization algorithm by ignoring the presence of the other robots in the environment. Following this approach, we could use a single robot localization algorithm in every robot in the team, but robots would ignore useful information available in their neighbors. For these reasons, in the literature, multi-robot localization problem is referred to as a collaborative/cooperative localization problem [21].

In this paper, we develop, analyze and study a localization strategy, in which robots neither share any movement constraint nor access to centralized information. This solution exploits self-organized cooperation in a swarm of robots to reduce each individual location error. In a nutshell, each robot's knowledge consists of an estimate of its own location and an associated confidence level that decreases with the distance travelled. This information is purely local and results from individual experience with the environment and other robots. In order to maximize its confidence level, each individual updates its estimates using the information available in its neighborhood. Hence, each individual will adopt the estimate of its immediate neighbors with a probability that increases with the ratio of confidence levels. This adaptive dynamics, previously defined as *Social*

Odometry [11], confers to each robot the possibility of gaining knowledge from others, by imitating estimated locations that offer higher confidence levels. Estimated locations, confidence levels and actual locations of the robots dynamically change in order to guide each robot to its goal. This simple online social odometry allows the population of robots to both reduce individuals errors and efficiently reach a common objective. Moreover, this collective navigation uses sucessfully minimal local communication, to promote an efficient collective performance and removes the need of stationary robots. We introduce the *Social Generalized Induced Kalman Filter* (SGIKF) and the *Social Generalized Induced Fermi Filter* (SGIFF) based on a resemblance to the *Kalman Filter* (KF) and its *Induced Kalman Filter* (IKF) derivation [9]. The filter development flows to the confidence level concept, inspired by the spectral norm of the covariance matrix.

For the algorithm validation, real localization and communication systems are needed to provide situated and abstract communication. Abstract communication refers to communication protocols in which only the content of the message carries a meaning and the physical signal (the medium) that transports the message does not have any semantic properties [29]. Differently, situated communication means that both the physical properties of the signal that transfers the message and the content of the message contribute to its meaning (see [6] for more details). For example, when an individual tells us to "come here" the content of the message has not all the information, but the location of the sender with respect to our position makes the message gets the meaning of "come to my position". One way to do so is to let the communicating robots extract from the signal the location of the communicating source. Therefore, these systems are commonly called localization and communication systems. Note that abstract communication tends to be used in multi-robot systems in which a wireless network provides the required structures to transmit messages from a specific sender to a specific receiver [33]. The use of wireless networks implies experiments with long range communication. Because of the number of robots and the dimensions of the experimental environments typically used, such communication span across the whole group of robots. In swarm robotics, this is in contradiction with the concept of locality which is seen as an important ingredient to achieve scalability. Therefore, in our experimental validation we make use of a local communication sensor [10] which offers situated and abstract communication to the robots.

2 Related work

Probabilistic methods have been applied with remarkable success to robot localization [27, 4, 3, 2, 12, 13, 5, 34]. Most of these approaches are based on Markov localization methods which make use of dead-reckoning and absolute or relative measurements. The key idea is that each robot maintains an estimate on its position which will be updated according to its odometry calculations and measurements in the environment. The most used probabilistic method has been the Kalman Filter (KF) [16, 17, 20]. It is an optimal filter that estimates a state vector containing the robot position and the parameters characterizing the odometry error introduced by the designer. The KF makes a number of assumptions on the system, measurements and different noises that are involved in the estimation problem. It presupposes that the system and measurements are adequately modeled by a linear dynamic system, and that noises are independent and white. Moreover, the KF presumes that the initial state of the system is also independent and Gaussian distributed.

Although the KF is an efficient recursive filter, it requires to introduce in the robots external information that models the environment and it is also computationally costly. For instance, in [30] robots navigate in an indoor environment where a map is given to the robots and make use of the KF. Each robot senses the environment, correct its measure according to the observation of others robots and exchange the information with the rest of the team. In this work, the way the information is exchanged is not presented. In [22], each robot measures its relative orientation and shares the information with the rest of the group. During the navigation cycle, each robot collects data from its proprioceptive sensors to perform the prediction step of KF while it shares information from the exteroceptive sensors with the rest of the team during the update. In [15]

a method based on a combination of maximum likelihood estimation and numerical optimization was introduced. This method allows the error on the robot localization to be reduced by using the information coming from relative observations among the robots in the team. A distributed multi-robot localization strategy was introduced based on an Extended KF (EKF) used to fuse proprioceptive and exteroceptive sensor data. In [26], the authors deal with the automatic learning of the spatial distribution given a set of images offered by all the members of the group, thanks to a centralized system which takes care of all the information. In [23], the robots are equipped with a compass which increases the efficiency of the filter. In [21] robots dynamically correct the estimation autonomously evaluated when a second robot comes across and they exchange their estimates. The algorithm combines the robustness of an EKF with its interlaced (IEKF) implementation [25]. Robots make use of a centralized system (wireless communication) for the data exchange. In [19] a new approach based on a series of EKFs hierarchically distributed is introduced. The team is broken down into several groups and, for each group, an EKF estimates the locations of all the members of the group in a local frame attached to one robot. However, this robot is specific and is considered as the group leader.

Other approaches for exploring the environment are based on map construction [7], which are also costly in computational terms. Most of these implementations build maps incrementally by iterating localization for each new reading of each sensor on a robot.

Different works have been implemented recently in the multi-robot localization approach based on local communication. In [8] robots navigate in an environment to which they know the map and improve their global localization each time they meet another robot exchanging their estimates. In [22] each of the robots collects sensor data regarding its own motion and shares this information with the rest of the team during the update cycles. A single estimator, in the form of a KF, processes the available positioning information from all the members of the team and produces a position estimate for every one of them. In [18] the authors extend the EKF approach presented in [22] by considering a general relative observation between two robots. In order to exploit the information contained in any relative observation between two robots, the authors derive the EKF equations to integrate a generic relative observation, based on the bearing, distance and orientation of another robot. In [21], the authors implement a collaborative localization system in simulation. When two robots are sufficiently close to communicate they exchange information about the robot pose and a fixed landmark. The data transmitted are used as "virtual" sensors for the receiving robot.

While being successful in achieving their goals, all these frameworks present several limitations: *i)* they are have high computational consumption as a result of optimal filters and map construction, *ii)* some robots are not allowed to move while others are tracking the distance between them, thus representing a misuse of resources, *iii)* robots must maintain visual contact with the rest of the group at all times, and *iv)* in some cases have to communicate with a central device to update or download maps, synchronize movements, or update positions.

The collective behavior proposed in this paper avoids the previous problems in a collective and self-organized manner.

3 The robot: Hardware and motion

3.1 Robot hardware

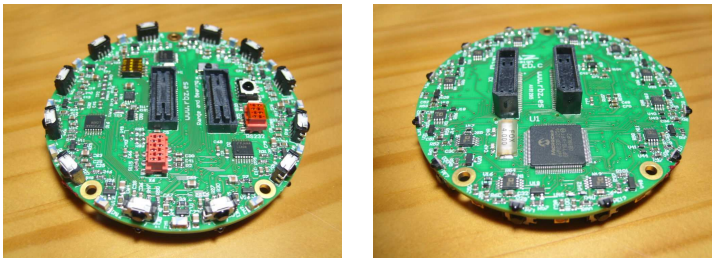
E-pucks are modular, robust and non-expensive robots designed by Francesco Mondada and Michael Bonani from *Ecole Polytechnique Fédérale de Lausanne* (EPFL) for research and educational purposes (see Figure 1). They are small wheeled cylindrical robots, 7 cm in diameter, equipped with a variety of sensors, and whose mobility is ensured by a differential drive system. The *e-puck* robot uses two miniature stepper motors with gear reduction. The motor has 20 steps per revolution and the gear has a reduction of 50:1. A circular ring is used as a tire for the wheel friction. The distance between the wheels is about 53 mm. The maximum speed of the wheels is about 1,000 steps/s, which corresponds to one wheel revolution per second.

Figure 1: *E-pucks* robots.

The *e-puck* hardware and software are fully open source providing low-level access to every electronic device and offering extension possibilities [1]. *E-pucks* are equipped with 8 infrared proximity sensors, a 3D accelerometer, a ring of 8 LEDs and a CMOS camera. Extension boards communicate with the main board through an I2C, SPI or RS232 bus. Finally, a Bluetooth communication is available for programming the robot and communicating data to a computer or to other Bluetooth devices.

3.2 Communication hardware

We have equipped each robot with a local communication board, the *E-puck Range & Bearing* board [10]. The board allows robots to communicate locally, obtaining at the same time both the range and the bearing of the emitter without the need of any centralized control or any external reference. The board relies on infrared communications with frequency modulation and is composed of two interconnected modules for data and power measurement. The *E-puck Range & Bearing* board is controlled by its own processor. Each board includes 12 sets of IR emission/reception modules (see Figure 2). Each of these modules is equipped with one infrared emitting diode, one infrared modulated receiver and one infrared photodiode[2]. The modules, as shown in Figure 3, are nearly uniformly distributed on the perimeter of the board; so, the distance between them is approximately 30°.



(a) (b)

Figure 2: (a) Top and (b) bottom view of the range and bearing board.

A Manchester code is implemented to allow any data sent at a certain distance to be received with the same intensity by the receiver. The implementation of the Manchester code allows a maximum data rate of 5 kbps. The range of communication can be software controlled from 0 cm to 80 cm.

[1] Further details on the robot platform can be found at http://www.e-puck.org.
[2] For an exhaustive description of the board see http://www.rbz.es/randb/.

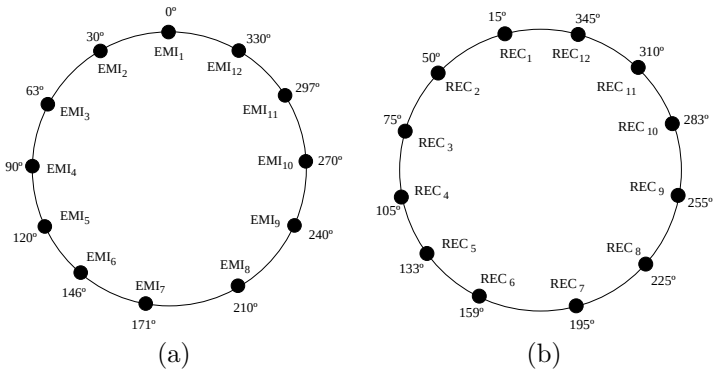


Figure 3: (a) Emitters and (b) receivers distribution around the perimeter of the board.

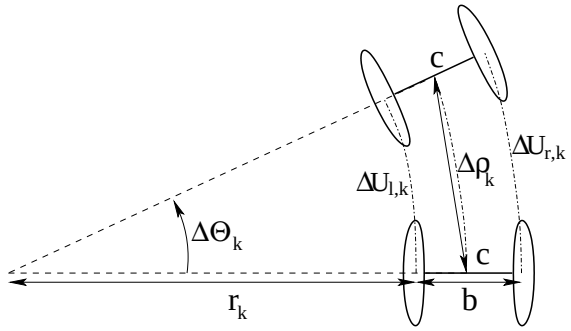


Figure 4: Path made by the wheels of a differential drive robot during a turn.

3.3 Robot motion

Let us assume a robot performs a movement in accordance with Figure 4, which can be expressed as the following trigonometric equations:

$$r_k \Delta\theta_k = \Delta U_{l,k} \tag{1}$$

$$(r_k + b)\,\Delta\theta_k = \Delta U_{r,k} \tag{2}$$

where b is the wheelbase of the robot, measured as the distance between the two ideal contact points between each wheel and the floor, r_k is the left wheel turning radius and $\Delta\theta_k$ is the turning angle at time k.

By subtracting Equation 1 from Equation 2 the turning angle is obtained:

$$\Delta\theta_k = (\Delta U_{r,k} - \Delta U_{l,k})/b \tag{3}$$

Adding Equation 1 and Equation 2 and substituting Equation 3 we obtain the linear movement of the center point c of the robot:

$$\Delta\rho_k = (\Delta U_{r,k} + \Delta U_{l,k})/2 \tag{4}$$

To obtain the robot's movement equation, let the location of a robot at time $k-1$ be:

$$\boldsymbol{x}_{k-1} = \begin{bmatrix} x_{k-1} & y_{k-1} & \theta_{k-1} \end{bmatrix}^T \tag{5}$$

where (x_{k-1}, y_{k-1}) are the Cartesian coordinates and θ_{k-1} is the orientation with respect to a global reference.

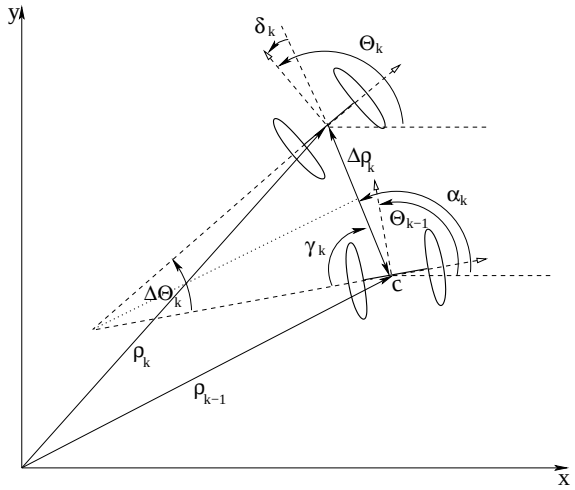


Figure 5: The robot's displacement on a plane.

A rotation $\Delta\theta_k$ and a translation $\Delta\rho_k$ move the robot to a new location $\boldsymbol{x}_k$ in accordance with Figure 5:

$$\boldsymbol{x}_k = \begin{bmatrix} x_{k-1} + \Delta\rho_k cos\left(\alpha_k\right) \\ y_{k-1} + \Delta\rho_k sin\left(\alpha_k\right) \\ \alpha_k + \delta_k \end{bmatrix} \tag{6}$$

If we approximate the arc movement of the midpoint of the robot for the chord ($\Delta\rho_k$) as shown in Figure 4, we observe that angle $\alpha_k = \pi - \gamma_k + (\theta_{k-1} - \pi/2)$, where $\gamma_k = \pi/2 - \Delta\theta_k/2$. Therefore $\alpha_k = \theta_{k-1} + \Delta\theta/2$. Moreover, $\theta_k = \alpha_k + \delta_k$, where $\delta_k = \pi/2 - \gamma_k$. Hence, $\theta_k = \theta_{k-1} + \Delta\theta_k$

By substituting these angles into Equation 6, we obtain the movement equation of the robot as [3]:

$$\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \begin{bmatrix} \Delta\rho_k cos(\theta_{k-1} + \Delta\theta_k/2) \\ \Delta\rho_k sin(\theta_{k-1} + \Delta\theta_k/2) \\ \Delta\theta_k \end{bmatrix} \tag{7}$$

Let us represent equation 7 as

$$\boldsymbol{x}_k = f\left(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, \boldsymbol{v}_k\right) \tag{8}$$

where $\boldsymbol{x}_{k-1}$ is the $\begin{bmatrix} x_{k-1} & y_{k-1} & \theta_{k-1} \end{bmatrix}^T$ state vector, $\boldsymbol{u}_k$ is the $\begin{bmatrix} \Delta U_{r,k} & \Delta U_{l,k} \end{bmatrix}^T$ input vector and $\boldsymbol{v}_k$ denotes the system noise. $\boldsymbol{v}_k \sim N\left(0, \boldsymbol{Q}\right)$ indicates a white noise with a zero mean and covariance matrix $\boldsymbol{Q}$, which models the uncertainties of the odometry model.

Looking at Equation 7 and assuming the system has no errors, the state vector is redefined as $\boldsymbol{x}_k = f\left(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, 0\right)$. For modelling the reliability of the localization measurement, we define $\boldsymbol{P}$ as the covariance matrix of our measure:

$$\boldsymbol{P}_k = \boldsymbol{A}_k \boldsymbol{P}_{k-1} \boldsymbol{A}_k^T + \boldsymbol{W}_k \boldsymbol{Q}_k \boldsymbol{W}_k^T \tag{9}$$

where $\boldsymbol{P}_0 = 0$ and $\boldsymbol{Q}_k$ is defined as:

$$\boldsymbol{Q}_k = \begin{bmatrix} k_r|\Delta U_{r,k}| & 0 \\ 0 & k_l|\Delta U_{l,k}| \end{bmatrix} \tag{10}$$

[3]Notice that this equation approximates the robot's movement and it is not exact. Nevertheless, we will use it as if the robot's movement was calculated as an infinitesimal displacement.

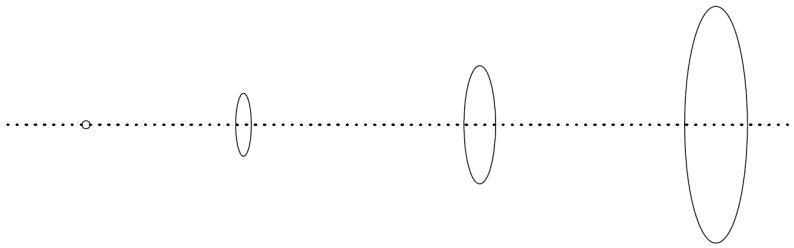


Figure 6: Non-systematic error propagation when a robot is travelling in a straight line.

where k_r and k_l are error constants representing the nondeterministic errors of the interaction of the floor and the right and left wheel respectively.

Finally, $\boldsymbol{A}_k$ and $\boldsymbol{W}_k$ are the Jacobians of $f(\cdot)$ with regard to $\boldsymbol{x}_{k-1}$ and $\boldsymbol{u}_k$, respectively:

$$\begin{aligned} \boldsymbol{A}_k &= \begin{bmatrix} 1 & 0 & -\Delta\rho_k sin\left(\zeta_k\right) \\ 0 & 1 & \Delta\rho_k cos\left(\zeta_k\right) \\ 0 & 0 & 1 \end{bmatrix} \\ \boldsymbol{W}_k &= \begin{bmatrix} \frac{1}{2}cos\left(\zeta_k\right) - \frac{\Delta\rho_k}{2b} sin\left(\zeta_k\right) & \frac{1}{2}cos\left(\zeta_k\right) + \frac{\Delta\rho_k}{2b} sin\left(\zeta_k\right) \\ \frac{1}{2}sin\left(\zeta_k\right) + \frac{\Delta\rho_k}{2b} cos\left(\zeta_k\right) & \frac{1}{2}sin\left(\zeta_k\right) - \frac{\Delta\rho_k}{2b} cos\left(\zeta_k\right) \\ \frac{1}{b} & -\frac{1}{b} \end{bmatrix} \end{aligned} \tag{11}$$

where $\zeta_k = \theta_{k-1} + \frac{\Delta\theta_k}{2}$ and $\Delta\theta_k$ and $\Delta\rho_k$ are defined by Equations 3 and 4.

Using the covariance matrix, the position estimation can be represented as an ellipsoid surrounding the robot position when no errors are computed. Projecting this ellipsoid in the $x-y$ plane, we model each computed robot position as a characteristic "error ellipse", which indicates a region of uncertainty for the actual position [28]. This region increases with the distance travelled (see Figure 6), until some absolute position measurement resets it.

4 Communication

4.1 The foraging task

We set up a foraging task which serves as an example for the algorithm derivation. Inside an arena we introduce two goal areas (i.e. nest and prey) which have to be located by the robots (See Figure 7a).

Once the areas have been located, the robots must forage from nest to prey endlessly. Robots use dead-reckoning methods to estimate and reach the nest and prey locations. When robot i finds the nest or the prey, it stores its *a priori* estimated location information as $\widehat{\boldsymbol{x}}^{nest,i}_{k|k-1}$ and $\widehat{\boldsymbol{x}}^{prey,i}_{k|k-1}$ respectively. Additionally, the robot keeps track of the path travelled since it left the nest or the prey denoted by $p^{nest,i}_{k|k-1}$ and $p^{prey,i}_{k|k-1}$ respectively, which represents the inverse of the *a priori* confidence level the robot has about its estimated information. (See Figure 7b).

At a given time step (k), robot i checks if there is another robot to communicate with. If there is not, it updates its *a posteriori* estimated goal locations and confidence levels as:

$$\begin{aligned} \widehat{\boldsymbol{x}}^{nest,i}_{k|k} &= \widehat{\boldsymbol{x}}^{nest,i}_{k|k-1} \\ p^{nest,i}_{k|k} &= p^{nest,i}_{k|k-1} \\ \widehat{\boldsymbol{x}}^{prey,i}_{k|k} &= \widehat{\boldsymbol{x}}^{prey,i}_{k|k-1} \\ p^{prey,i}_{k|k} &= p^{prey,i}_{k|k-1} \end{aligned} \tag{12}$$

In the next step $(k+1)$, the inverse of the *a priori* estimated confidence levels are updated with the distance travelled (Δd^i_{k+1}) in the time step duration:

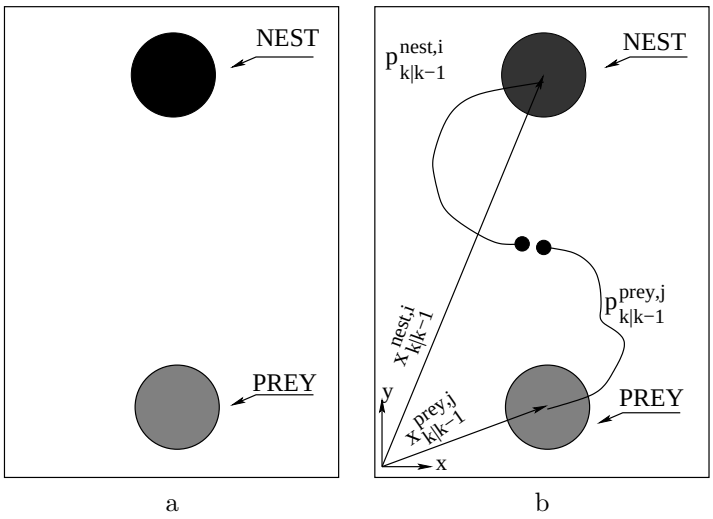


Figure 7: (a) Arena for a simple foraging task. (b) Robot information on the nest and prey areas and their confidence levels.

$$\begin{aligned} p^{nest,i}_{k+1|k} &= p^{nest,i}_{k|k} + \Delta d^i_{k+1} \\ p^{prey,i}_{k+1|k} &= p^{prey,i}_{k|k} + \Delta d^i_{k+1} \end{aligned} \tag{13}$$

Therefore, if no encounter between the robots is produced, the confidence level decreases endlessly until the robot arrives at the nest or prey, resetting it or until it gets lost.

If two robots meet, they communicate and update their estimates. In what follows we show all the different nest location exchange options, which are equivalent for the prey exchange.

- **None of the two robots know the nest location:** Robots do not exchange any information.
- **One robot knows the nest location:** Let us assume that robot i is the one who has previously visited the nest and has an estimated location $\widehat{\boldsymbol{x}}^{nest,i}_{k|k-1}$ and its correspondent confidence level $p^{nest,i}_{k|k-1}$. Hence, robot j updates its information as the one provided by robot i:

$$\begin{aligned} \widehat{\boldsymbol{x}}^{nest,j}_{k|k} &= \widehat{\boldsymbol{x}}^{nest,i}_{k|k-1} \\ p^{nest,j}_{k|k} &= p^{nest,i}_{k|k-1} \end{aligned} \tag{14}$$

- **Both robots know the nest location:** In this case, both robots exchange their information according to:

$$\begin{aligned} \widehat{\boldsymbol{x}}^{nest,i}_{k|k} &= \boldsymbol{f}_1\left(\widehat{\boldsymbol{x}}^{nest,i}_{k|k-1}, \widehat{\boldsymbol{x}}^{nest,j}_{k|k-1}, p^{nest,i}_{k|k-1}, p^{nest,j}_{k|k-1}\right) \\ p^{nest,i}_{k|k} &= \boldsymbol{f}_2\left(p^{nest,i}_{k|k-1}, p^{nest,j}_{k|k-1}\right) \end{aligned} \tag{15}$$

and

$$\begin{aligned} \widehat{\boldsymbol{x}}^{nest,j}_{k|k} &= \boldsymbol{f}_1\left(\widehat{\boldsymbol{x}}^{nest,j}_{k|k-1}, \widehat{\boldsymbol{x}}^{nest,i}_{k|k-1}, p^{nest,j}_{k|k-1}, p^{nest,i}_{k|k-1}\right) \\ p^{nest,j}_{k|k} &= \boldsymbol{f}_2\left(p^{nest,j}_{k|k-1}, p^{nest,i}_{k|k-1}\right) \end{aligned} \tag{16}$$

where $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$ are the update functions discussed in Section 4.3. We will show that thanks to $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$, when two robots update their information, they end up with the same estimated knowledge and confidence level:

$$\begin{array}{l} \widehat{\boldsymbol{x}}_{k|k}^{nest,i} = \widehat{\boldsymbol{x}}_{k|k}^{nest,j} \\ p_{k|k}^{nest,i} = p_{k|k}^{nest,j} \end{array} \tag{17}$$

4.2 The Social Induced Kalman Filter

We assume that a robot i acts as a sensor for robot j and vice versa. When the robots communicate, they transmit their *a priori* state vector and *a priori* covariance matrix. If we define this communication following the next equations:

$$\begin{array}{l} \boldsymbol{R}_k^i = \boldsymbol{P}_{k|k-1}^j \\ \boldsymbol{R}_k^j = \boldsymbol{P}_{k|k-1}^i \end{array} \tag{18}$$

$$\boldsymbol{y}_k = h\left(\boldsymbol{x}_k, \boldsymbol{v}_k\right) = \boldsymbol{x}_k + \boldsymbol{v}_k \tag{19}$$

where $\boldsymbol{P}_{k|k-1}$ is the *a priori* covariance matrix and the random variable $\boldsymbol{v}_k \sim N\left(0, \boldsymbol{R}\right)$ denotes a white noise, with 0 mean and associated covariance matrix $\boldsymbol{R}$. It can be demonstrated [9] that:

$$\boldsymbol{K}_k^i + \boldsymbol{K}_k^j = \boldsymbol{I} \tag{20}$$

$$\boldsymbol{P}_{k|k}^i = \boldsymbol{P}_{k|k}^j \tag{21}$$

where $\boldsymbol{K}$ is the KF gain.

Therefore, the KF equation correction phase can be denoted as:

$$\boldsymbol{K}_k^i = \boldsymbol{P}_{k|k-1}^i \left(\boldsymbol{P}_{k|k-1}^i + \boldsymbol{P}_{k|k-1}^j\right)^{-1} \tag{22}$$

$$\widehat{\boldsymbol{x}}_{k|k}^i = \boldsymbol{K}_k^j \widehat{\boldsymbol{x}}_{k|k-1}^i + \boldsymbol{K}_k^i \left(\widehat{\boldsymbol{x}}_{k|k-1}^j + \boldsymbol{x}_k^{ij}\right) \tag{23}$$

$$\boldsymbol{P}_{k|k}^i = \boldsymbol{K}_k^j \boldsymbol{P}_{k|k-1}^i \tag{24}$$

Given the previous KF equations, we call *Social Induced Kalman Filter* (SIKF) to the one which substitutes the *a priori* and *a posteriori* covariances matrices and the Kalman gain by its induced norms [9]. If $||x||$ is a vector norm, we define the induced matrix norm as $||\boldsymbol{A}|| = max\left(||Ax|| \,\forall\, ||x|| = 1\right)$. Therefore, the SIKF is defined by the *a posteriori* state vector equation:

$$\widehat{\boldsymbol{x}}_{k|k} = \widehat{\boldsymbol{x}}_{k|k-1} + ||\boldsymbol{K}_k|| \left(\boldsymbol{y}_k - h\left(\widehat{\boldsymbol{x}}_{k|k-1}, 0\right)\right) \tag{25}$$

In what follows we will use the spectral norm as induced norm. The spectral norm is defined as the square root of the largest eigenvalue of the semidefinite positive matrix $\boldsymbol{A}^*\boldsymbol{A}$, where $\boldsymbol{A}^*$ represents the conjugate transpose matrix of $\boldsymbol{A}$:

$$||\boldsymbol{A}||_2 = \sqrt{\lambda_{max}\left(\boldsymbol{A}^*\boldsymbol{A}\right)} \tag{26}$$

It is easy to prove that if $\boldsymbol{A}$ is a real symmetric matrix, the spectral norm follows the following equations:

$$||\boldsymbol{A}||_2 = \lambda_{max}\left(\boldsymbol{A}\right) \tag{27}$$

$$||\boldsymbol{I} + \boldsymbol{A}||_2 = 1 + ||\boldsymbol{A}||_2 \tag{28}$$

and if $\boldsymbol{A}$ and $\boldsymbol{B}$ are two invertible matrices, then:

$$||\boldsymbol{AB}||_2 = ||\boldsymbol{A}||_2 \, ||\boldsymbol{B}||_2 \tag{29}$$

Therefore, if we define $g_k = ||\boldsymbol{K}_k||_2$ and $p = ||\boldsymbol{P}||_2$, where the matrices are symmetric and positive definite we obtain:

$$g_k^i + g_k^j = 1 \tag{30}$$

Then, we rewrite the KF equations based on the previous assumptions to obtain the SIKF equations:

$$\widehat{\boldsymbol{x}}_{k|k-1}^i = f\left(\boldsymbol{x}_{k-1|k-1}^i, \boldsymbol{u}_{k-1}^i, 0\right) \tag{31}$$

$$p_{k|k-1}^i = \left|\left|\boldsymbol{A}_k \boldsymbol{P}_{k-1|k-1}^i \boldsymbol{A}_k^T + \boldsymbol{V}_k \boldsymbol{Q}_{k-1} \boldsymbol{V}_k^T\right|\right| \tag{32}$$

$$g_k^i = \frac{p_{k|k-1}^i}{p_{k|k-1}^i + p_{k|k-1}^j} \tag{33}$$

$$\widehat{\boldsymbol{x}}_{k|k}^i = \left(1 - g_k^i\right) \widehat{\boldsymbol{x}}_{k|k-1}^i + g_k^i \left(\widehat{\boldsymbol{x}}_{k|k-1}^j + \boldsymbol{x}_k^{ij}\right) \tag{34}$$

$$p_{k|k}^i = \left(1 - g_k^i\right) p_{k|k-1}^i \tag{35}$$

4.3 Social Odometry

Based on the SIKF, we define its generalized form (*Social Generalized Induced Kalman Filter* (SGIKF)) as the filter which substitutes the induced norm by any other scalar value. In *Social Odometry* we define this value as the inverse of the confidence level a robot has about its measurements. Approaching a real application and inspired by social insects, we define the inverse of the confidence level as the distance traveled by a robot since it left a specific area (d_k^i).

Following these principles, two different filters are presented:

- *Social Generalized Induced Kalman Filter (SGIKF):* It follows the SIKF equations, where the inverse of the confidence level is represented by d_k^i.
- *Social Generalized Induced Fermi Filter (SGIFF):* A modification of the SIKF where g_k is modified according to a sigmoid function and the inverse of the confidence level is also represented by d_k^i.

In both filters, when a robot encounters a neighbor, it transmits its estimated location and confidence level. Hence, each robot has the opportunity to adopt the estimates of other robots present in their neighborhood.

4.3.1 Social Generalized Induced Kalman Filter (SGIKF)

Since the spectral norm of the covariance matrix $\boldsymbol{P}$ grows endlessly until a communication is established or the robots arrive at one of the goals, we define the inverse of the *a priori* confidence level ($p_{k|k-1}^i$) of robot i as the distance travelled (d_k^i) since the robot left a specific area. Therefore the prediction stage for the induced covariance matrix is defined as:

$$p_{k|k-1}^i = d_k^i \tag{36}$$

This implementation allows the robot not to calculate the covariance matrix at each time step, and therefore to save computational time.

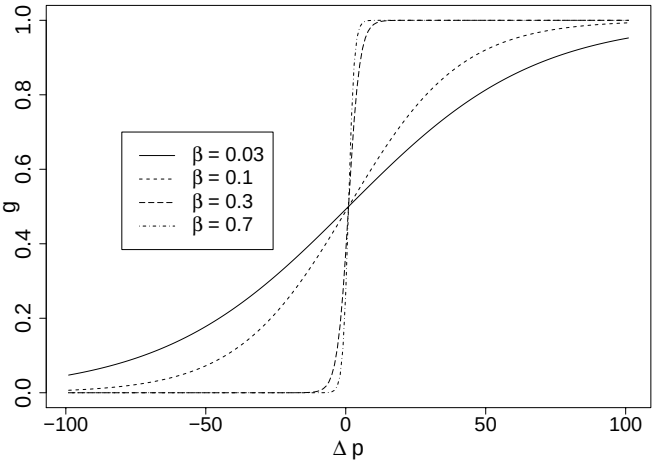


Figure 8: The Fermi function which allows the robots to decide between their own estimate and the information provided by the others.

4.3.2 Social Generalized Induced Fermi Filter (SGIFF)

In the SGIFF, the inverse of the *a priori* confidence level, represented by $p_{k|k-1}$, is also calculated as the distance travelled since the robot left a specific area.

To calculate g_k^i, we adopt the so called pairwise comparison rule [31, 24, 32] often adopted in evolutionary/social dynamic studies, to code the social learning dynamics, which makes use of the Fermi distribution (see also Figure 8):

$$g_k^i = \frac{1}{1 + e^{-\beta(\Delta p_{k|k-1})}} \tag{37}$$

where $\Delta p_{k|k-1} = p_{k|k-1}^i - p_{k|k-1}^j$ and β measures the importance of the relative confidence levels in the decision making. For low values of β, the decision making proceeds by ignoring the confidence levels whereas for high value of β, we obtain a pure imitation dynamics commonly used in cultural evolution [14] defined by a sharp step function. In the first case, the confidence level works as a small perturbation to a simple average between the two estimates, while in the latter, each robot is ready to completely ignore the estimate which has a smaller relative confidence level.

Hence, we use a weighted average to obtain the new location $\widehat{\boldsymbol{x}}_{k|k}^i$ and confidence level $p_{k|k}^i$ using the Fermi function:

$$\widehat{\boldsymbol{x}}_{k|k}^i = \left(1 - g_k^i\right) \widehat{\boldsymbol{x}}_{k|k-1}^i + g_k^i \left(\widehat{\boldsymbol{x}}_{k|k-1}^j + \boldsymbol{x}_k^{ij}\right) \tag{38}$$

$$p_{k|k}^i = \left(1 - g_k^i\right) p_{k|k-1}^i + g_k^i p_{k|k-1}^j \tag{39}$$

A comparison chart about the IKF, SGIKF and SGIFF equations is shown in Table 1. In the SGIFF, we observe the *a posteriori* confidence level of robot i adds a terminus based on the confidence level of robot j ($g_k^i p_{k|k-1}^j$). This is because, following a social dynamics approach, we want to offer a more rational than statistical approach to the SGIFF. When two robots communicate, they implicitly agree on a middle point between both confidence levels. Therefore, when a communication is established, the robot with a worst confidence level will improve it and the one with the better confidence level will reduce it.

Table 1: Comparison table between the Social Induced Kalman Filter and Social Odometry equations.

	Social Induced Kalman Filter	Social Odometry Filters
$\widehat{\boldsymbol{x}}^i_{k\|k-1}$	$f\left(\boldsymbol{x}^i_{k-1\|k-1}, \boldsymbol{u}^i_{k-1}, 0\right)$	$f\left(\boldsymbol{x}^i_{k-1\|k-1}, \boldsymbol{u}^i_{k-1}, 0\right)$
$p^i_{k\|k-1}$	$\left\|\boldsymbol{A}_k \boldsymbol{P}^i_{k-1\|k-1} \boldsymbol{A}^T_k + \boldsymbol{V}_k \boldsymbol{Q}_{k-1} \boldsymbol{V}^T_k\right\|$	Δd^i_k
g^i_k	$\frac{p^i_{k\|k-1}}{p^i_{k\|k-1}+p^j_{k\|k-1}}$	SGIKF: $\frac{p^i_{k\|k-1}}{p^i_{k\|k-1}+p^j_{k\|k-1}}$ SGIFF: $\frac{1}{1+e^{-\beta\left(p^i_{k\|k-1}-p^j_{k\|k-1}\right)}}$
$\widehat{\boldsymbol{x}}^i_{k\|k}$	$\left(1-g^i_k\right)\widehat{\boldsymbol{x}}^i_{k\|k-1} + g^i_k\left(\widehat{\boldsymbol{x}}^j_{k\|k-1} + \boldsymbol{x}^{ij}_k\right)$	$\left(1-g^i_k\right)\widehat{\boldsymbol{x}}^i_{k\|k-1} + g^i_k\left(\widehat{\boldsymbol{x}}^j_{k\|k-1} + \boldsymbol{x}^{ij}_k\right)$
$p^i_{k\|k} i$	$\left(1-g^i_k\right)p^i_{k\|k-1}$	SGIKF: $\left(1-g^i_k\right)p^i_{k\|k-1}$ SGIFF: $\left(1-g^i_k\right)p^i_{k\|k-1} + g^i_k p^j_{k\|k-1}$

5 Control architecture

We have designed the robots' controllers based on a behavior-based architecture. The behavior is implemented following a *Subsumption* architecture [1]. At the lowest level, each behavior is represented using an augmented finite state machine (AFSM). Each AFSM performs an action and is responsible for its world perception. In the following we describe two architectures: one with and one without communication. For each of them, we first give a high-level description and then detail the Subsumption architecture used, the behaviors and the conditions that inhibit/suppress each behavior.

In the controllers, robots make use of the following inputs and outputs:

- *Clock*: the system clock runs at 100 ms. Some of the AFSMs need the clock input to perform the odometry movement calculations.
- *Infrared Sensors:* 8 infrared sensors are distributed around the the robot's perimeter (see Figure 9). They are used to detect the presence of obstacles and to detect any neighbors with whom the robots can communicate.
- *Ground sensors:* 3 infrared sensors are located in the front-bottom part of the robot (see Figure 9). Robots differentiate the areas depending on the color of the ground. The nest is represented as black, prey as grey and the rest of the arena as white.
- *RANDB Emitter and Receiver:* the emission and reception module of the *E-puck Range & Bearing* board.
- *Motors:* the two differential drive motors (see Figure 9).

5.1 Controller without communication

The robots are initially located at random positions inside a fixed area in the center of the arena. Inside this area robots do not perceive the central place (nest) or the resource site (prey) areas. Once a robot finds the nest or the prey, it stores its position and continues with a random walk until it finds the other area. When both areas have been located, the robots try to go from one area to the other endlessly. Because of the movement errors, robots might arrive at some coordinates that they wrongly estimate inside the area. At this point a robot considers itself lost, it resets its estimated locations and starts carrying out a random walk until it finds both areas again. If a robot correctly arrives at one of the areas, it stores the new position coordinates.

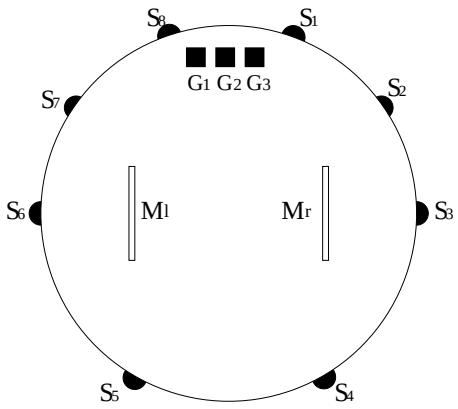


Figure 9: Depiction of an *e-puck*. S_i $i \in [1, 8]$ refer to the infrared sensors used as IR proximity sensors. G_i $i \in [1, 3]$ refer to the ground sensors. M_l and M_r are the left and right motor respectively.

Figure 10 shows the architecture diagram of the controller. Each layer corresponds to a robot behavior and arrows connecting the different AFSMs are the connector, inhibitor, suppressor or reset signals.

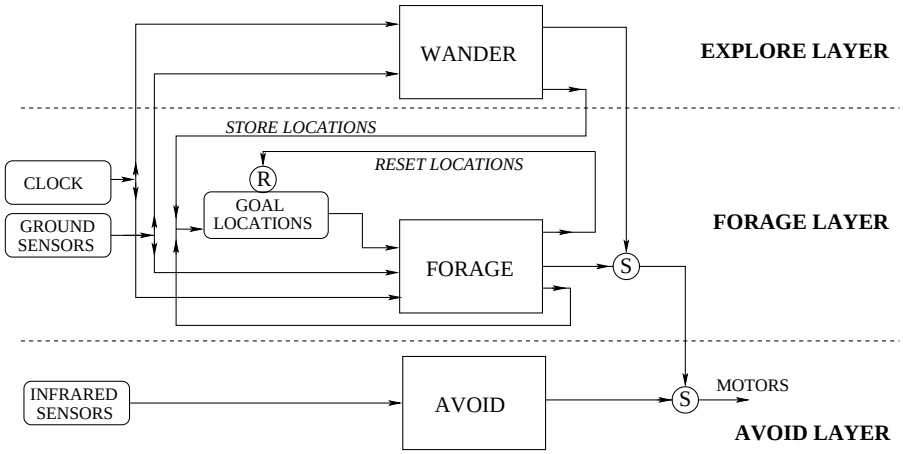


Figure 10: The behavior structure of the foraging agents for a controller without communication. The inhibitors are represented by S, the reset by R.

The AFSMs are described in the following:

- *Avoid*: the state machine return a vector taking into account all the IR sensors above a threshold. The direction sent to the motors is opposite to the vector.
- *Forage*: the robot has location information of both the nest and the prey area. It walks from nest to prey following the shortest path. If the robot arrives at one of both areas, detected by the ground sensors, it stores the new estimated position and goes towards the other area. If the robot arrives at a place where the area was supposed to be but does not perceive it, it resets the *Goal Locations* memory. This last action gets the robot lost and it starts wandering again.
- *Wander*: the robot carries out a random walk. If a nest or prey area is found, the robot stores its position in the *Goal Locations* memory.

All the layers are permanently active. The upper layer (explore layer) always outputs values and it is suppressed if the forage layer outputs signals to the motors when both goal locations are known. If an obstacle is detected, the avoid layer inhibits the previous layers' outputs and obtains the motor control.

5.2 Controller with communication

In this controller, the robots are initially located as in the controller without communication. Robots maintain the *Avoid*, *Forage* and *Wander* AFSMs unchanged. When two robots encounter they exchange their estimates about the goal areas. The information is exchanged and updated following the filters presented in Section 4.3.

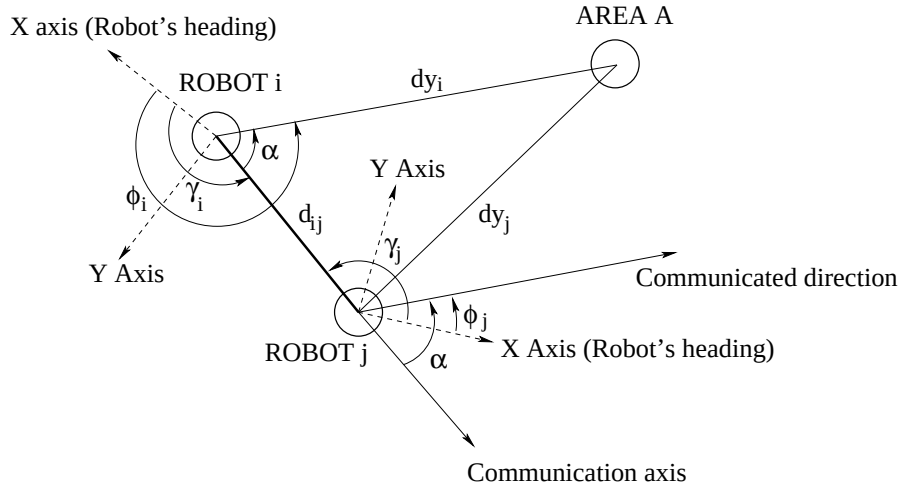


Figure 11: Robots sharing information on their estimated location.

However, robots do not share a global coordinates system, so they rely only on the communication axis to transform the information transmitted by their neighbor into their own frame. Figure 11 shows an example of how information about the estimated location of Area Y, previously visited by robot j, is transmitted from robot j to robot i. In a first step, robot j transmits its estimate of the distance dy^j and direction ϕ^j of area Y to robot i. For the direction, the value transmitted is the angle α, obtained from ϕ^j using the communication beam as reference axis: $\alpha = \phi^j - \gamma^j$. In a second step, robot i transforms the received data into its own coordinates system. First, it calculates the direction pointed out by robot j as $\phi^i = \gamma^i + \alpha - \pi$, followed by the calculation of the location $\left(\widetilde{x}^j, \widetilde{y}^j\right)$ of area Y. This location is represented by $\widehat{\boldsymbol{x}}^j_{k|k} + \boldsymbol{x}^{ij}_k$ in the *Social Odometry* filter equations (see Section 4.3). This computation takes into account the transformation to its own frame of reference and robot i computes robot j information in the following way:

$$\begin{aligned} \widetilde{x}^j &= |x^{ij}| cos\left(\gamma^i\right) + dy^i cos\left(\phi^i\right) \\ \widetilde{y}^j &= |x^{ij}| sin\left(\gamma^i\right) + dy^i sin\left(\phi^i\right) \end{aligned} \tag{40}$$

Figure 12 shows the architecture diagram of the controller. Each layer corresponds to a robot behavior and arrows connecting the different AFSMs, the inhibitor, suppressor or reset signals. In addition to the architecture used for the controller without communication, two more AFSMs are required for this controller:

- *Receive Data:* the robot translates the information received into its own reference axis.
- *Send Data:* the robot transforms the information to communicate according to the common reference axis (communication axis), and sends it to its neighbors.

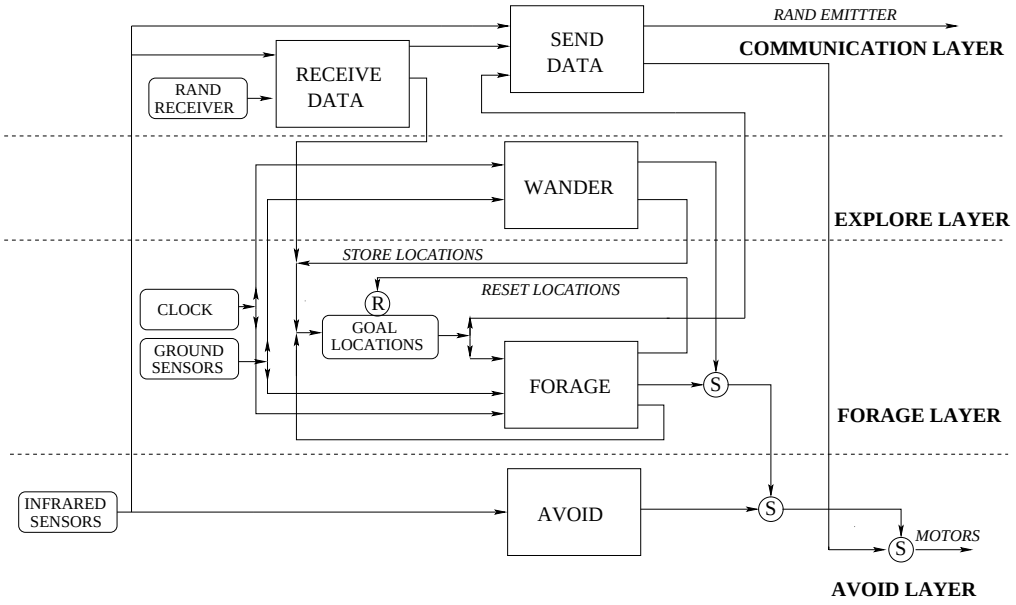


Figure 12: The behavior structure of the foraging agents for a controller with communication. The inhibitors are represented by S, the reset by R.

The architecture follows the same suppression and inhibition principles as the one without communication. However, in this architecture the communication layer takes control of the motors' output in case something is received in the RANDB receiver. When the robot receives an input from the *E-puck Range & Bearing* board, it stops the movement, tries to communicate with its neighbor and release the motors control. This strategy is adopted to stabilize the range and bearing measure and therefore to avoid the robots moving while they are communicating.

6 Experimental evaluation

In the following, we report experimental results with simulated and real robots in which the outcomes of the controllers using communication strategies and the one not using communication are compared.

6.1 Experiments in simulation

In our simulation, an *e-puck* is modelled as a cylindrical body of 3.5 cm in radius that holds 8 infrared sensors distributed around the body, 3 ground sensors on the bottom-front part of the body and a range and bearing communication sensor. A differential drive system composed of two wheels is fixed to the body of the simulated robot. For the three types of sensors, we have sampled real robot measurements and mapped the data into the simulator. Furthermore, we introduce errors to simulate the standard deviation of the different sensors. $\pm 20\,\%$ noise is added to the infrared sensors and $\pm 30\,\%$ to the ground sensors. In the range and bearing sensor, noise is added to the bearing ($\pm 20\,^\circ$) and range (± 2.5 cm) values. Moreover, each message emitted can be lost with a probability that varies linearly from 1 % when the sender-receiver distance is less than 1 cm, to 50 % when the two robots are 15 cm from each other.

Three different experimental setups (ESs) have been chosen to study the convergence and comparison of the different algorithms. All setups are carried out in a similar arena (see Figure 7) where the dimensions and number of robots are changed as detailed in Table 2.

For each experimental setup we have tested five different communication strategies:

Table 2: Description of the three simulated experimental setups for the single objective localization experiment.

	ES1	ES2	ES3
Area dimensions (m^2)	1.2 x 1.7	3.0 x 4.25	6 x 8.5
Initial area radius (m)	0.2	0.5	0.5
Nest and prey radius (m)	0.1	0.1	0.1
Number of robots	10	50	100
Experiment duration (s)	1800	7200	7200

- *No communication (NC)*: Robots do not communicate.
- *Single SGIKF (SSGIKF)*: Robots use the SGIKF to correct the information received related to the area they are going to.
- *Double SGIKF (DSGIKF)*: Robots use the SGIKF to correct the information received related to both areas.
- *Single SGIFF (SSGIFF)*: Robots use the SGIFF to correct the information received related to the area they are going to.
- *Double SGIFF (DSGIFF)*: Robots use the SGIFF to correct the information received related to both areas.

Each experimental setup has been repeated 30 times. The performance of the robots in the foraging task under study is measured as the number of total round trips completed from the central place to the resource site and back during the experiment.

6.1.1 Recruitment process

We first study the time it takes for all the robots to locate the nest and prey areas. When robots do not communicate they are exploring the environment until they find both areas. Each robot uses the same algorithm, but they ignore useful information available from their neighbors (i.e. other robots that have already found one or both areas). In any of the other four strategies, when robots are looking for one of the areas and they find a neighbor, they stop and check if the neighbor is transmitting information about the unknown area. In this case, they take the neighbor's estimates as their own and go towards the communicated estimate.

Figure 13 shows the time it takes for all the robots to visit both areas at least once. We observe how the convergence on the path speeds up with the communication strategies. A robot, which has found one or both areas, is able to recruit other robots. All the communication strategies perform approximately in the same way. The recruitment time efficiency speeds up four times on the communication strategies compared to the NC for the ES1 and the ES2. Moreover, note than in the ES3 not all the robots with a NC controller are able to find the path within the 7,200 s run.

6.1.2 Retrieval process

We have also tested the benefits of the different communication strategies in a retrieval process. We define the retrieval process as the task in which robots have to transport "virtual" items from the prey to the nest area. Each time a robot completes a run from the nest to the prey and comes back to the nest, we consider the robot has succeeded in its task and count one round trip. We have tested the different communication strategies in the three ESs. In any of the communication strategies, robots show a better performance than the NC. In the NC strategy, robots rely only on their estimates and once a robot has lost the correct location of its goal, it has to explore the environment and find the areas by chance, which explains the poor performance of this strategy.

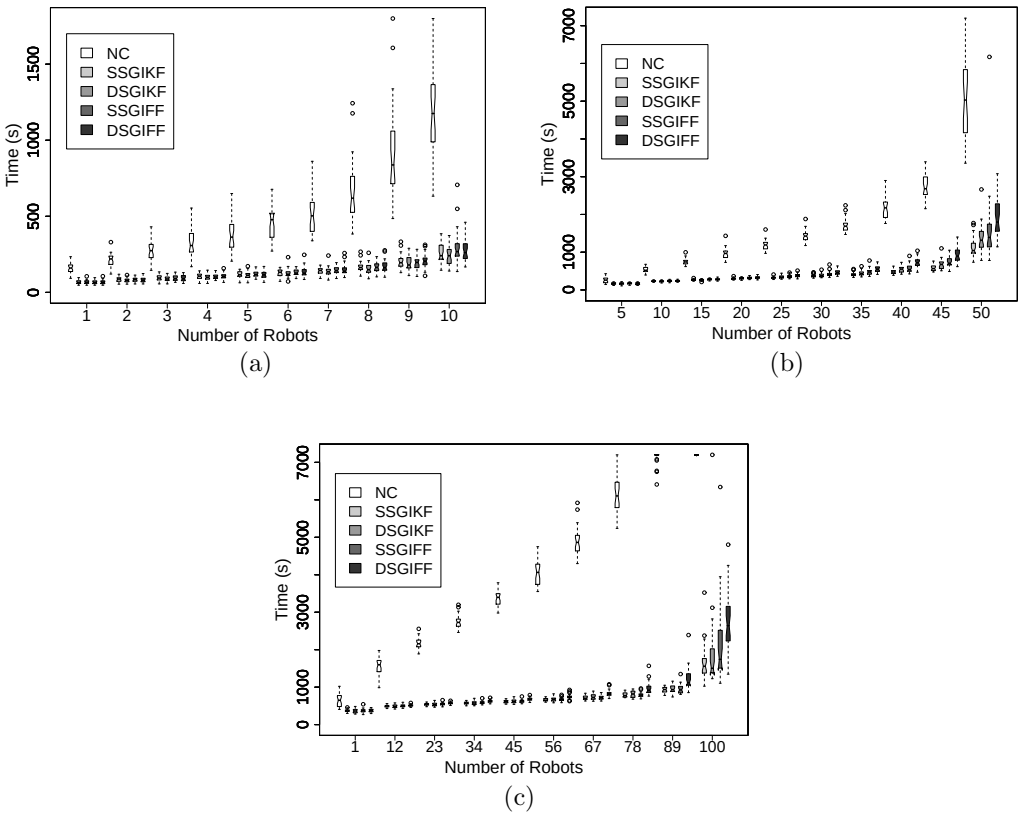


Figure 13: Recruitment results for the five communication strategies for the (a) ES1 (b) ES2 and (c) ES3, in a single objective scenario. (30 replications for each boxplot). Each box comprises observations ranging from the first to the third quartile. The median is indicated by a horizontal bar, dividing the box into the upper and lower part. The whiskers extend to the farthest data points that are within 1.5 times the interquartile range. Outliers are shown as circles.

Figure 14 shows results for the two SGIKFs compared to the NC. We observe that the SGIKFs increase 2.5 times the performance of the retrieval process with respect the NC for the ES1. For the other two ESs, robots without communication are not able to perform more than 50 successful retrievals while the SGIKFs achieve 1400 and 2000 retrievals in average for the ES2 and the ES3 respectively. We observe that the single implementation of the filter, when robots only communicate their estimate about the goal they are aiming at, performs better than the double one. This is because when robots are moving from one area to the other, they have better information on the goal they are coming from than the one they are aiming at. When a robot i encounters another neighbor (robot j), this last one is typically coming from the place robot i is aiming at. Therefore, robot i improves its goal estimate with the more reliable information of robot j (the estimate on where it is coming from). If robot j communicates information about the goal it is aiming at, as in the double filter implementation, it is disrupting the information from robot i which will be propagated to the rest of the neighbors robot i encounters later.

Figure 15 shows the same experiments for the SGIFFs with different β parameters. We have defined 8 values where $\beta \in [10^{-5}, 100]$. A maximum retrieval process is achieved when $\beta = 10^{-2}$ for the ES1 and ES2, and $\beta = 10^{-3}$ for the ES3. Finally, note that the best βs perform better than in the SGIKFs, while any of the other β perform worse than the SGIKF implementations.

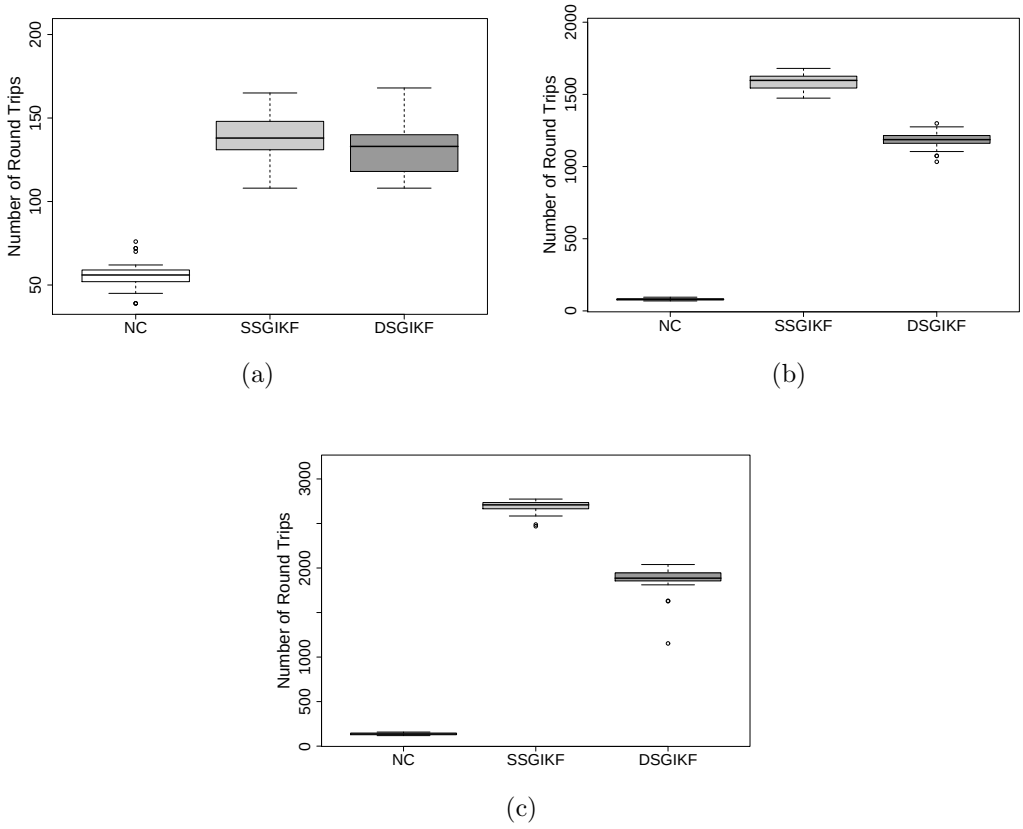


Figure 14: Retrieval results comparison between the NC, SSGIKF and DSGIKF for the (a) ES1 (b) ES2 and (c) ES3, in a single objective scenario. (30 replications for each boxplot).

6.1.3 Stability

In any of the communication strategies we observe, not all of the robots are able to keep themselves on the path. When many robots try to enter the nest and prey areas at the very same time or when there is not enough robots on the path to communicate with, the robots get lost. However, when robots get lost they are typically close to the nest or prey area. Therefore, a lost robot that uses a communication strategy is likely to meet another robot from whom it gets its estimates and has a high probability of finding the path again. Figure 16 shows the average of the robots on the path for the three experimental setups. In all the communication strategies, 50-70 % of the robots (depending on the filter used) are always in the path at steady state. However, no more than 20 % of the total number of robots are able to stay on the path with the NC strategy.

6.2 Experiments with real robots

Real experiments are carried out in a 1.7x1.2 m^2 arena, during 1800 s with 10 *e-pucks*. Robots start in a central round area of 0.2 cm radius in random positions and orientations. Data collection is managed through the Bluetooth connection. Due to the Bluetooth limitations, two different computers are used to communicate with 5 different robots each. Robots are initialized at the same time thanks to a standard TV remote control. Additionally, robots keep track of a timer which is initialized with the remote control and allows the robots to have the same time reference. When a robot arrives at one of the two goals (i.e. nest or prey) or it gets lost, the robot sends

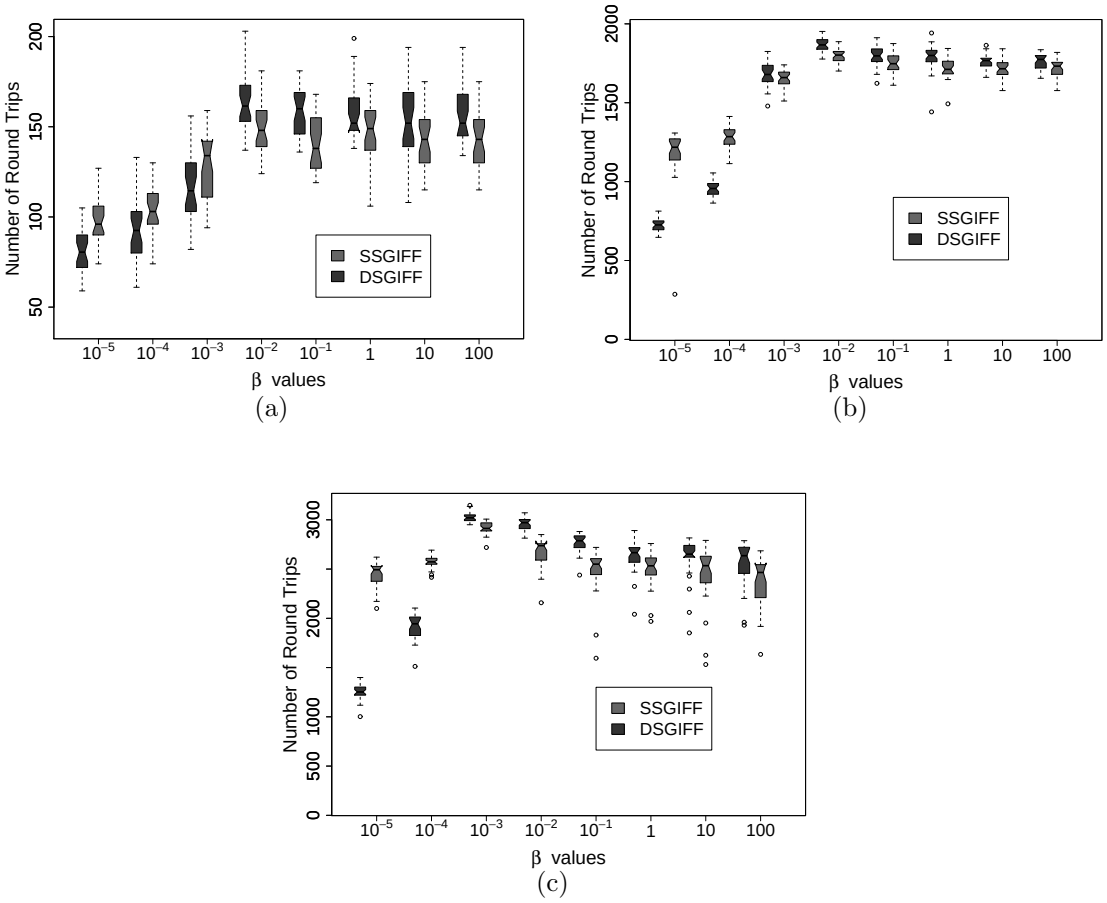


Figure 15: Retrieval results for different β values of the SGIFFs for the (a) ES1 (b) ES2 and (c) ES3, in a single objective scenario. (30 replications for each boxplot).

a Bluetooth command to the computer, indicating the state in which it finds itself (i.e. nest, prey or lost) and the time at which it has been produced. After 1800 s the controller stops and robots must be randomly initialized again. The 8 IR sensors are used as input to the avoid and communication behaviors. The nest and prey areas are detected with the ground sensors. The communication range has been limited to 15 cm to avoid the IR signals to spread over the arena, as in the simulation experiments.

We have tested the 5 different communication strategies defined in Section 6.1. For the SGIKFs, only the best β value obtained in simulation (i.e. $\beta = 10^{-2}$) has been programed. Finally, each experimental setup was repeated 30 times to allow for statistical comparisons.

In Figure 17 we observe the recruitment process achieved by the real robots. Compared to the simulation results we notice a similar behavior and the same improvement in average. However, the standard deviation grows because of communication imperfections discussed later.

Figure 18 shows the retrieval process comparison between the 5 different strategies. We observe a small reduction in the number of round trips. In average 120 round trips for the SGIKFs compared to the 150 obtained in simulation and 140 instead of 170 for the SGIFFs. Therefore there is only a difference of 20% between real and simulated experiments. However, robots with any of the communication strategies are able to carry out in average twice more retrievals than with the NC strategy.

Finally, Figure 19 shows the stability of the robots on the path. A larger oscillation is seen with

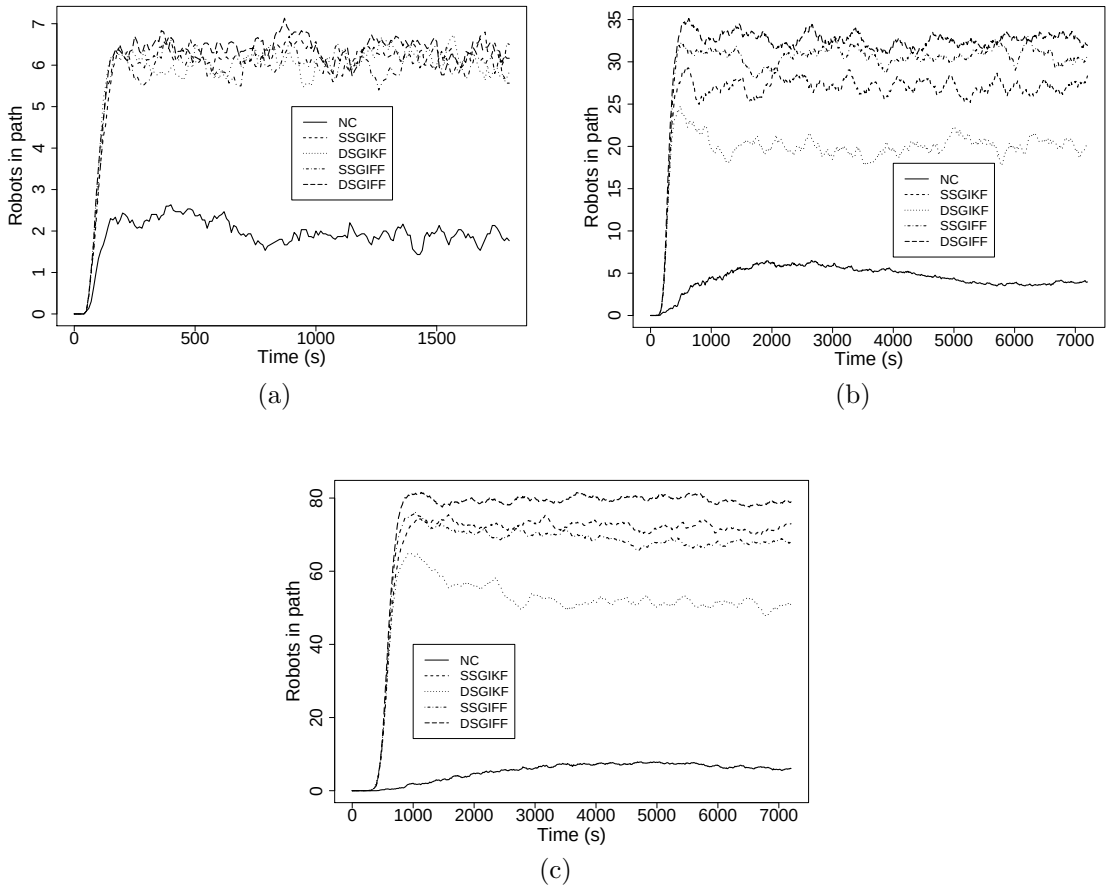


Figure 16: Stability results comparison for the five communication strategies for the (a) ES1 (b) ES2 and (c) ES3, in a single objective scenario.

respect to the simulated experiments. For the communication strategies, we observe an average of 5 robots in the path compared to the 6 robots obtained in simulation.

The differences observed between the real and simulated experiments are mainly due to the working of the range and bearing sensor. These differences arise from the imperfection of the communication model implemented in simulation. Another problem is the reflection obtained from the borders of the arena, which distort some of the bearing measures when robots are near those borders. Moreover, we have observed some interferences between the range and bearing sensor and the IR proximity sensors. These alterations in the range and bearing produce extra errors not modelled in the simulation, which make the robots miscalculate the information given by other neighbors. However, we observe the same behavior in all the qualitative measures and only a decrease of 10-20 % in the quantitative ones.

Moreover, the time used in the communication is longer in the real robots scenario. This is because errors in the communication makes the communication layer repeat some messages. Therefore, robots wait a little more time (around 20-30%) in the communication process, which explains the reduction of the number of "virtual" items retrieved in the experiments with real robots.

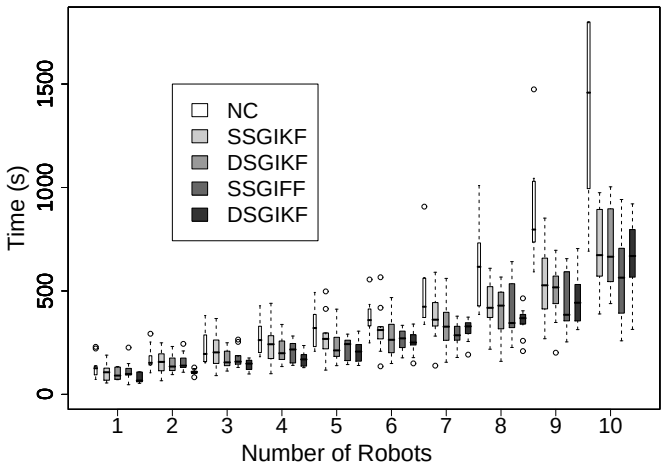


Figure 17: Recruitment process for 10 real *e-pucks* in a 1.7x1.2 m^2 arena.

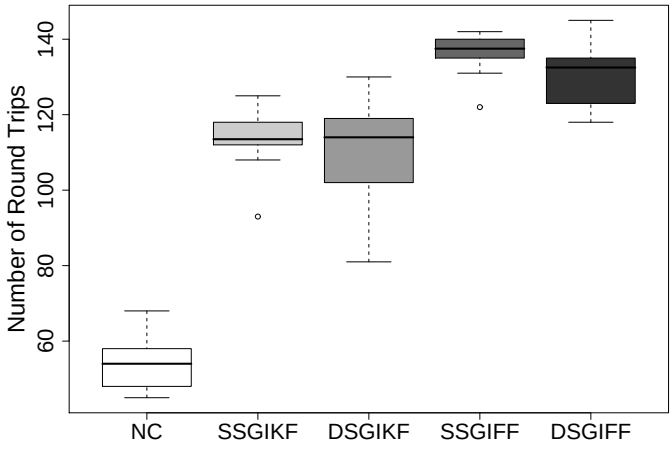


Figure 18: Retrieval process for 10 real *e-pucks* in a 1.7x1.2 m^2 arena.

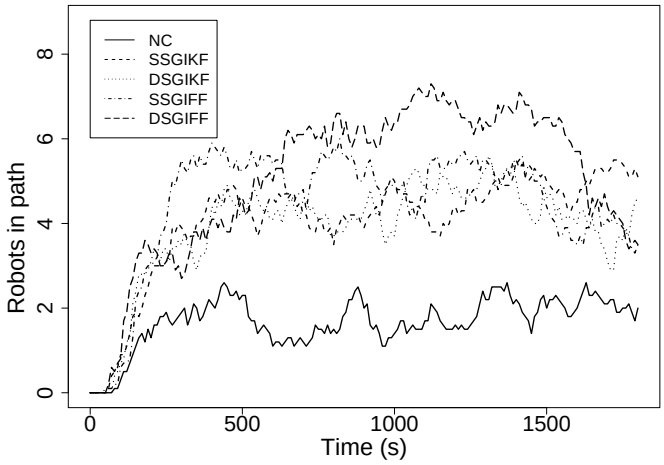


Figure 19: Stability process for 10 real *e-pucks* in a 1.7x1.2 m^2 arena.

7 Conclusions

In this paper we have described a social localization strategy in which robots use pairwise local communication to share knowledge about specific locations to improve their performance in a foraging task. By letting the robots use the estimates of others, we engineer an efficient and decentralized knowledge sharing mechanism which allows the robots to achieve their goals, both from an individual and group perspective. This simple mechanism drives the system to a successful collective pattern improving the individuals' behavior.

We have also compared two different *Social Odometry* strategies: the Social Generalized Induced Kalman Filter, a simplification of the Kalman Filter equations and the Social Generalized Induced Fermi Filter as an imitation based dynamic algorithm. Our experiments have demonstrated that complex behaviors can result from simple local interactions based on simple behavior-based controllers such as the subsumption architecture. A simple foraging task has been created as test-bed for the algorithm test. Local communication allows the robots to improve drastically the group performance and shows it to be stable. Additionally, we observe that the algorithms implement an implicit recruitment process that speeds up the initial exploration phase mandatory to achieve the foraging task.

Finally, we would like to stress that the performance of the *Social Odometry* filters allows an optimistic forecast concerning the use of online self-organized methodologies in the field of swarm robotics. Online methodologies, from communication to evolution, may represent the missing aspect necessary for truly adaptive populations of robots.

Acknowledgements

A. Campo and M.Dorigo acknowledge support from the Belgian F.R.S.-FNRS. This work was partially supported by the Spanish *Ministerio de Educación y Ciencia*, within the *Plan Nacional de I+D+I 2007-2010. (Gestión de la Demanda Eléctrica Doméstica con Energía Solar Fotovoltaica, ENE2007-66135)* and the *ANTS* project, an *Action de Recherche Concerté* funded by the Scientific Research Directorate of the French Community of Belgium.

References

[1] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Jorunal of Robotics and Automation*, 2(1):14–23, 1986.

[2] W. Burgard, A. Derr, D. Fox, and A. B. Cremers. Integrating gglobal position estimation and position tracking for mobile robots: The dynamic markov localization aapproach. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1–6, Piscataway, NJ, 1998. IEEE Press.

[3] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 896–901, Cambridge, MA, 1996. AAAI Press/MIT Press.

[4] A. R. Cassandra, L. P. Kaelbling, and J. A. Kurien. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 963–972, Piscataway, NJ, 1996. IEEE Press.

[5] K. Chong and L. Kleeman. Accurate odometry and error modelling for a mobile robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2783–2788, Piscataway, NJ, 1997. IEEE Press.

[6] W. J. Clancey. *Situated Cognition: On Human Knowledge and Computer Representations.* Cambridge University Presss, Cambridge, UK, 1997.

[7] G. Dudek and P. Mackenzie. Model-based map construction for robot localization. In *Proceedings of Vision Interface*, pages 97–102, Toronto, ON, 1993. CIPPR Society Press.

[8] D. Fox, W. Gurgard, H. Kruppa, and S. Thrun. Collaborative multi-robot exploration. *Autonomous Robots*, 8(3):325–344, 2000.

[9] A. Gutiérrez. *Social Odometry: Distributed Location Knowledge for Swarm Robotics Based on Local Communication.* PhD thesis, Escuela Técnica Superior de Ingenieros de Telecomunicación - Universidad Politécnica de Madrid, 2009.

[10] Alvaro Gutiérrez, Alexandre Campo, Marco Dorigo, Daniel Amor, Luis Magdalena, and Félix Monasterio-Huelin. An open localization and local communication embodied sensor. *Sensors*, 8(11):7545–7563, 2008.

[11] Alvaro Gutiérrez, Alexandre Campo, Francisco C. Santos, Carlo Pinciroli, and Marco Dorigo. Social odometry in populations of autonomous robots. In Marco Dorigo, Mauro Birattari, Christian Blum, Maurice Clerc, Thomas Stützle, and Alan F.T. Winfield, editors, *Ant Colony Optimization and Swarm Intelligence, 6th International Conference, ANTS 2008, Brussels, Belgium, September 2008, Proceedings*, volume LNCS 5217 of *Lecture Notes in Computer Science*, pages 371–378. Springer-Verlag, Berlin, Germany, 2008.

[12] J-S. Gutmann, T. Weigel, and B. Nebel. Fast, accurate and robust self-localization in polygonal environments. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1412–1419, Piscataway, NJ, 1999. IEEE Press.

[13] J.S. Gutmann, T. Weigel, and B. Nebel. A fast, accurate, and robust method for self-localization in polygonal environments using laser-rangefinders. *Advanced Robotics Journal*, 14(1):1–17, 2001.

[14] P. Hammerstein. *Genetic and Cultural Evolution of Cooperation.* MIT Press, Cambridge, MA, 2003.

[15] A. Howard, M.J. Matarić, and G.S. Sukhatme. Localization for mobile robot teams using maximum likelihood estimation. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 434–459, Piscataway, NJ, 2002. IEEE Press.

[16] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[17] T.D. Larsen, M. Bak, N.A. Andersen, and O. Ravn. Location estimation for autonomously guided vehicle using an augmented Kalman filter to autocalibrate the odometry. In *FUSION 98 Spie Conference*, pages 33–39, Las Vegas, NV, 1998. CSREA Press.

[18] A. Martinelli. Multi-robot localization using relative observations. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2808–2813, Piscataway, NJ, April 2005. IEEE Press.

[19] A. Martinelli. Improving the precision on multi robot localization by using a series of filters hierarchically distributed. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1053–1058, Piscataway, NJ, October 2007. IEEE Press.

[20] A. Martinelli and R. Siegwart. Estimating the odometry error of a mobile robot during navigation. In *Proceedings of the 1st European Conference on Mobile Robots*, pages 218–223, Warszawa, Poland, September 2003. Zturek Research Scientific Inst. Press.

[21] S. Panzieri, F. Pascucci, and R. Setola. Multirobot localisation using interlaced extended kalman filter. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 2816–2821, Piscataway, NJ, 2006. IEEE Press.

[22] S.I. Roumeliotis and G.A. Bekey. Distributed multirobot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–794, 2002.

[23] S.I. Roumeliotis and I.M. Rekleitis. Propagation of uncertainty in cooperative multirobot localization: Analysis and experimental results. *Autonomous Robots*, 17(1):41–54, 2004.

[24] F. C. Santos, J. M. Pacheco, and T. Lenaerts. Cooperation prevails when individuals adjust their social ties. *PLoS Computational Biology*, 2(10):e140, 2006.

[25] L. Glielmo R. Setola and F. Vasca. An interlaced extended kalman filter. *IEEE Transactions on Automatic Control*, 44(8):1546–1549, 1999.

[26] R. Sim and G. Dudek. Self-organizing visual maps. In *Proceedings of the National Conference on Artificial Intelligence*, pages 470–475, Cambridge, MA, July 2004. MIT Press.

[27] R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1080 – 1087, San Mateo, CA, July 1995. Morgan Kaufmann.

[28] R. C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainly. *International Journal of Robotics Research*, 5(4):56–68, 1987.

[29] K. Støy. Using situated communication in distributed autonomous mobile robots. In *Proc. of the 7^{th} Scandinavian Conf. on artificial intelligence*, pages 44–52, Amsterdam, NL, 2001. IOS Press.

[30] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 321–328, Piscataway, NJ, April 2000. IEEE Press.

[31] A. Traulsen, M.A. Nowak, and J.M. Pacheco. Stochastic dynamics of invasion and fixation. *Physical Review - Serie E*, 74(1):011909.1–011909.5, July 2006.

[32] A. Traulsen, J.M. Pacheco, and M.A. Nowak. Pairwise comparison and selection temperature in evolutionary game dynamics. *Journal of Theoretical Biology*, 246(3):522–529, June 2007.

[33] A. Tubaishat and S. Madria. Sensor networks: An overview. *IEEE Potentials*, 22(2):20–23, 2003.

[34] C. Ming Wang. Location estimation and uncertainty analysis for mobile robots. *Autonomous Robot Vehicles*, 1(1):1230–1235, 1988.