Applied Soft Computing 43 (2016) 489–497

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A swarm-based approach to real-time 3D indoor localization: Experimental performance analysis☆

Stefania Monica[a,*], Gianluigi Ferrari[b]

[a] Department of Mathematics and Computer Science, University of Parma, I-43124 Parma, Italy
[b] Wireless Ad-hoc and Sensor Networks Laboratory (WASNLab), Department of Information Engineering, University of Parma, I-43124 Parma, Italy

ARTICLE INFO

Article history:
Received 13 July 2015
Received in revised form 18 January 2016
Accepted 13 February 2016
Available online 10 March 2016

Keywords:
Particle swarm optimization (PSO)
UWB localization
P410 RCMs

ABSTRACT

In this paper, the problem of indoor localization in wireless networks is addressed relying on a swarm-based approach. We assume to know the positions of a few number of sensor nodes, denoted as anchor nodes (ANs), and we aim at finding the position of a target node (TN) on the basis of the estimated distances between each AN and the considered TN. Since ultra wide band (UWB) technology is particularly suited for localization purposes (owing to its remarkable time resolution), we consider a network composed of UWB devices. More precisely, we carry out an experimental investigation using the PulsOn 410 ranging and communication modules (RCMs) produced by time domain. Using four of them as ANs and one of them as TN, various topologies are considered in order to evaluate the accuracy of the proposed swarm-based localization approach, which relies on the pairwise (AN-TN) distances estimated by the RCMs. Then, we investigate how the accuracy of the proposed localization algorithm changes if we apply to the distance estimates a recently proposed stochastic correction, which is designed to reduce the distance estimation error. Our experimental results show that a good accuracy is obtained in all the considered scenarios, especially when applying the proposed swarm-based localization algorithm to the stochastically corrected distances. The obtained results are satisfying also in terms of software execution time, making the proposed approach applicable to real-time dynamic localization problems.

1. Introduction

The interest in wireless networks keeps on growing, since, due to the availability of small, cheap, and intelligent electronic components, they represent a scalable and low-cost solution to several practical challenges. In a wireless network, nodes may be equipped with one or more sensors which can measure physical quantities such as: temperature, speed, acceleration, and pressure. The variety of properties of the environment which can be monitored allows dealing with a large number of applications, such as: smart homes; environmental monitoring; medical monitoring of patients in hospitals; military surveillance; and localization and tracking of people and vehicles [1].

This paper is focused on the application of wireless networks to accurately localize targets in indoor environments. The main idea of the considered problem is that range measurements between a given number of nodes with known positions, denoted as anchor nodes (ANs), and a target node (TN) need to be used to estimate the position of the TN. This problem, which has many applications, is very interesting and, at the same time, challenging. As a matter of fact, indoor communications are particularly delicate since the presence of obstacles may cause non-line-of-sight (NLoS) and multipath propagation. In order to reduce the impact of these phenomena, we perform localization using ultra wide band (UWB) technology, which is particularly suited for various reasons. First of all, due to their large bandwidth, UWB systems transmit pulses whose duration is typically on the order of a nanoseconds. This guarantees a low probability that pulses received via multiple paths overlap and, therefore, the time of flight (ToF) of signals traveling between nodes can be accurately estimated. As a consequence, precise distance estimates between pairs of sensors and high localization accuracy can be obtained [2]. Moreover, the low duty cycle which characterizes UWB systems guarantees low energy consumption. Finally, due to the large frequency spectrum of UWB signals, they can penetrate through obstacles, which can easily be

☆ This paper is an extended, improved version of the paper *A swarm intelligent approach to 3D distance-based indoor UWB localization* presented at EvoComNet2015 and published in: Applications of Evolutionary Computing, Proceedings of 18th European Conference, EvoApplications 2015, Copenhagen, Denmark, April 8–10, 2015, LNCS 9028, pp. 91–102, Springer, 2015.

* Corresponding author. Tel.: +39 0521906900.
E-mail address: stefania.monica@studenti.unipr.it (S. Monica).

http://dx.doi.org/10.1016/j.asoc.2016.02.020
1568-4946/

found in indoor environments [3]. These aspects make UWB technology a leading candidate for accurate, low-cost, and low-power positioning systems.

Given the distance estimates between nodes, different kinds of localization algorithms have been proposed in the literature. The majority of these algorithms are based on the received signal strength (RSS) or the ToF of the signals traveling between nodes. The first class of methods relies on the knowledge of a relation between the received power of the transmitted signal and the distance between transmitter and receiver. The path-loss during propagation is characterized by the Friis formula, which reads (in the logarithmic domain) [3]:

$$\bar{P}(d) = P_0 - 10\beta \log_{10} \frac{d}{d_0} \quad d \geq d_0 \tag{1}$$

where P_0 is the (known) power (dimension: [dBm]) at the reference distance d_0 (dimension: [m]); β is the path-loss exponent (adimensional); and $\bar{P}(d)$ is the average received power (dimension: [dBm]) at distance d (dimension: [m]). From (1), it can be observed that the distance between two nodes can be estimated from the RSS measured at the receiver node, under the assumption that the transmitted signal energy is known. The accuracy of RSS-based localization relies heavily on channel knowledge, which is a strong requirement and a great challenge as well, especially in indoor scenarios.

Time-based positioning techniques rely on measurements of the ToFs of signals traveling between pairs of nodes. If two synchronized nodes communicate, the node receiving the signal can determine the time of arrival (ToA) of the incoming signal and, from the timestamp of the sending node, the ToF. If the nodes are not synchronized, time difference of arrival (TDoA) techniques can be employed, based on the estimation of the difference between the arrival times of UWB signals traveling back and forth between two nodes. Since, as discussed above, the ToF between signals can be accurately estimated with the UWB technology, in the following we consider ToF approaches.

Many ToF-based localization algorithms have been studied. Among them, it is worth recalling: (i) iterative methods, based on Taylor series expansion [4] or the steepest-descent algorithm [5], which guarantee fast convergence only for an initial estimate value close to the true solution (often difficult to obtain in real applications); (ii) closed-form methods, such as the Plane Intersection (PI) algorithm [6] and the Two-Stage Maximum-Likelihood (TSML) algorithm [7]. In particular, the TSML algorithm can attain the Cramer-Rao lower bound and, therefore, is typically considered as optimal [8]. Despite some good properties, these "geometrical" methods can lead to wrong position estimates in some particular topologies. As a matter of fact, they all deal with linear or non-linear systems of equations which can become ill-conditioned – this happens, for example, when the ANs lay on the same line or plane. In order to avoid these phenomena, different techniques have been considered, among which: (i) probabilistic methods based on Monte-Carlo simulations [9]; (ii) machine learning approaches [10]; (iii) and swarm-based techniques [11], [12].

Many of the results found in the literature are relative to theoretical bounds for accurate localization and rely on simulation results. In [13], for instance, a three-dimensional localization scenario is simulated and the performance obtained using the TSML algorithm is compared with that obtained using a swarm-based approach. In the current paper, we extend the preliminary investigation of [13]: in particular, we present performance results obtained using an experimental UWB testbed. More precisely, we make use of the PulsOn 410 Ranging and Communication Modules (RCMs) produced by Time Domain. Using these sensors, we consider different 3D localization scenarios inside a room and we use the range estimates produced by the RCMs to perform swarm-based localization. Moreover, we show how the performance of the proposed algorithms can be improved by using a recently proposed statistical correction to the range estimates. We also focus on the software execution time needed to perform localization, showing that the proposed approach is applicable also to dynamic localization.

This paper is organized as follows. In Section 2, the proposed swarm-based localization algorithm is derived. In Section 3, the considered experimental setup is described and different localization scenarios are described. In Section 4, the performance of the proposed algorithm is experimentally evaluated in terms of (i) localization accuracy and (ii) execution time needed to perform localization. Section 5 concludes the paper.

2. Localization algorithms

The proposed localization approach is based on the particle swarm optimization (PSO) technique [14]. We assume to have M ANs placed in an indoor environment and we aim at using distance estimates between each AN and the TN to localize the latter. We remark that $M \geq 4$, since three-dimensional localization requires at least 4 ANs [20]. We denote the ANs' coordinates as

$$\boldsymbol{s}_i = [x_i, y_i, z_i]^T \quad i \in \{1, \ldots, M\} \tag{2}$$

and we denote as $\{K_i\}_{i=1}^M$ the square of the Euclidean norm of $\{\boldsymbol{s}_i\}_{i=1}^M$, namely:

$$K_i = x_i^2 + y_i^2 + z_i^2 \quad i \in \{1, \ldots, M\}. \tag{3}$$

Moreover, we denote as $\boldsymbol{u} = [x, y, z]^T$ the true position (which is supposed to be unknown) of the TN which needs to be localized. Its estimated position will be denoted as $\hat{\boldsymbol{u}} = [\hat{x}, \hat{y}, \hat{z}]^T$. Using this notation, the true and estimated distances between the i-th AN and the considered TN are, respectively:

$$\begin{aligned} r_i &= ||\boldsymbol{u} - \boldsymbol{s}_i|| \quad i \in \{1, \ldots, M\} \\ \hat{r}_i &= ||\hat{\boldsymbol{u}} - \hat{\boldsymbol{s}}_i|| \quad i \in \{1, \ldots, M\}. \end{aligned} \tag{4}$$

Before explaining the swarm-based approach, let us make some geometric considerations which are the basis to perform localization. Given the (known) positions of the M ANs and the true distances between them and the currently considered TN, the position of the latter can be determined by intersecting the spheres centered in $\{\boldsymbol{s}_i\}_{i=1}^M$ with radii $\{r_i\}_{i=1}^M$, namely by solving the following system:

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 = r_1^2 \\ \ldots \\ (x - x_M)^2 + (y - y_M)^2 + (z - z_M)^2 = r_M^2. \end{cases} \tag{5}$$

The system of equations (5) is a quadratic system of M equations in the three unknowns x, y, and z and admits a unique solution, which corresponds to the true TN position.

Since the true distances $\{r_i\}_{i=1}^M$ between the ANs and the TN are (obviously) unavailable, in (5) we replace their coordinates with those of their estimates $\{\hat{r}_i\}_{i=1}^M$. Therefore, instead of considering (5), localization is performed on the basis of the following quadratic system:

$$\begin{cases} (\hat{x} - x_1)^2 + (\hat{y} - y_1)^2 + (\hat{z} - z_1)^2 = \hat{r}_1^2 \\ \ldots \\ (\hat{x} - x_M)^2 + (\hat{y} - y_M)^2 + (\hat{z} - z_M)^2 = \hat{r}_M^2. \end{cases} \tag{6}$$

Due to range errors, the circumferences associated with the M equations in (6) will not intersect in the same point as when solving (5). Therefore, the unknowns of the system (6) are denoted as $\hat{x}$, $\hat{y}$, and $\hat{z}$. A proper localization algorithm needs to be considered in order to derive the TN position estimate $\hat{\boldsymbol{u}}$.

Observe that the system (6) can be re-written in matrix notation:

$$\boldsymbol{i}\,\hat{\boldsymbol{u}}^T\hat{\boldsymbol{u}} + A\,\hat{\boldsymbol{u}} = \hat{\boldsymbol{k}} \tag{7}$$

where $\boldsymbol{i}$ is a $M \times 1$ vector with all elements equal to 1; $\hat{\boldsymbol{k}}$ is a $M \times 1$ vector whose $i-$th element is $\hat{r}_i^2 - K_i$; and A is the following $M \times 3$ matrix:

$$A \triangleq -2\begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_M & y_M & z_M \end{pmatrix}. \tag{8}$$

Possible algorithms to solve (7) can be found in [7]. Such algorithms are based on the least square technique, on the Taylor series expansion, or on maximum-likelihood methods. All these techniques suffer from ill-conditioning in the presence of specific nodes' configurations.

A more robust solution of an over-determinated system, such as (7), can be found by re-interpreting it as an optimization problem. In [7], the use of Genetic Algorithms (GA) is also proposed. In this paper, instead, we propose a PSO-based approach to solve (7). More precisely, the minimization problem associated with (7) can be written as follows:

$$\hat{\boldsymbol{u}} = \text{argmin}_{\boldsymbol{u}} F(\boldsymbol{u}) \tag{9}$$

where the fitness function $F(\boldsymbol{u})$ is defined as:

$$F(\boldsymbol{u}) \triangleq ||\hat{\boldsymbol{k}} - (\boldsymbol{i}\,\hat{\boldsymbol{u}}^T\hat{\boldsymbol{u}} + A\,\hat{\boldsymbol{u}})||.$$

The PSO algorithm was introduced in [14] as a global optimization algorithm. It is an iterative method where the set of potential solutions of the considered optimization problem corresponds to a swarm of particles. Each particle is associated with a position and a velocity, which are both updated at each iteration according to proper laws, in order to move the particles toward the optimal solution. At the beginning, the positions of all particles are randomly initialized in the search space with values $\boldsymbol{x}^{(i)}(0)$, where $i \in \mathcal{S}$ denotes the generic particle index and $\mathcal{S}$ is the set of indices which identify the particles. Similarly, the initial velocities are randomly initialized with values $\boldsymbol{v}^{(i)}(0)$. At any iteration $t \in \mathbb{N}$, each particle in the swarm is associated with a position $\boldsymbol{x}^{(i)}(t)$ and with a velocity $\boldsymbol{v}^{(i)}(t)$, which are updated according to proper rules meant to simulate "social" interactions between individuals [15]. More precisely, in the most general formulation of the PSO algorithm the velocity of particle i is updated, at each iteration, according to the following rule [16]:

$$\begin{aligned}\boldsymbol{v}^{(i)}(t+1) \;=\; & \omega(t)\boldsymbol{v}^{(i)}(t) + c_1R_1(t)(\boldsymbol{y}^{(i)}(t) - \boldsymbol{x}^{(i)}(t)) \\ & + c_2R_2(t)(\boldsymbol{y}(t) - \boldsymbol{x}^{(i)}(t)) \quad i \in \{1,\ldots,S\}\end{aligned} \tag{10}$$

where $S = |\mathcal{S}|$ denotes the number of particles in the swarm. Observe that the velocity of a particle at the iteration $t+1$ depends on the velocity at the previous iteration t through the first addend at the right-hand side of (10). More precisely, the value of the velocity of the i-th particle $\boldsymbol{v}^{(i)}(t)$ at the previous iteration is weighed according to the *inertial factor* $\omega(t)$. Typically, the inertial factor $\omega(t)$ is chosen as a decreasing function of t, in order to guarantee low dependence of the solution on the initial population and to reduce the exploration ability of the swarm as the number of iterations increases, making the method more simliar to a local search in the last iterations [16]. In (10), the parameters c_1 and c_2 are positive real parameters denoted as *cognition* and *social* parameters, respectively, while $R_1(t)$ and $R_2(t)$ are random variables uniformly distributed in (0, 1). To be more precise, the second addend at the right-hand side of (10) is meant to "push" each particle $i \in \{1,\ldots,S\}$ to its best position reached so far, denoted as $\boldsymbol{y}^{(i)}(t)$. Since the considered optimization problem is a minimization one, the best position is the one which corresponds to the lowest value of the fitness function and, therefore, can be expressed as:

$$\boldsymbol{y}^{(i)}(t) = \text{argmin}_{\boldsymbol{z} \in \{\boldsymbol{x}^{(i)}(0),\ldots\boldsymbol{x}^{(i)}(t)\}} F(\boldsymbol{z}). \tag{11}$$

The third addend at the right-hand side of (10), instead, aims at pushing each particle $i \in \{1,\ldots,S\}$ toward the global best position $\boldsymbol{y}(t)$ reached so far, which is defined as:

$$\boldsymbol{y}(t) = \text{argmin}_{\boldsymbol{z} \in \{\boldsymbol{y}^{(1)}(t),\ldots\boldsymbol{y}^{(S)}(t)\}} F(\boldsymbol{z}). \tag{12}$$

Observe that $\boldsymbol{y}(t)$ is the position which corresponds to the smallest value of the fitness function among all those reached by any particle in the swarm [15].

The definition of the velocities of the particle in the swarm given in (10) is then used to update the positions of the particles, according to the following rule:

$$\boldsymbol{x}^{(i)}(t+1) = \boldsymbol{x}^{(i)}(t) + \boldsymbol{v}^{(i)}(t+1) \quad i \in \{1,\ldots,S\}. \tag{13}$$

The iterative process which updates the positions of the particles in the swarm is repeated until a stopping condition is met. Possible stopping conditions for the PSO algorithm are the achievement of a given (maximum) number of iterations or a satisfying (sufficiently low) value of the fitness function. At the end of the iterative process, the solution is the position of the particle with the smallest value of the fitness function. Since (9) is a minimization problem, the solution is associated with the particle which best suits the optimization requirements in the last iteration.

The PSO algorithm is then used to solve the considered localization problem, as formulated in (9). In order to obtain the experimental results shown in this paper, we consider the following parameters. First, observe that while the PSO algorithm is typically used to address global optimization problems, here we use it to solve the convex minimization problem, defined in (9), which has only one minimum. Therefore, the "exploration abilities" of the PSO algorithm, which are typically useful to avoid local minima in global optimization problems, can be reduced. For this reason, we ignore the inertial factor, namely we set $\omega(t) = 0\ \forall t$ (as already proposed in [17]). Moreover, we also set $c_1 = 0$ in (10) in order to annihilate also the second addend at the right-hand side of (10). According to this choice, the only effect of the velocity is to push the particles toward the global best position. As a consequence, the convergence to the optimal solution is faster. Moreover, this choice of $\omega(t)$ and c_1 allows reducing the computational complexity of the algorithm. As a matter of fact, from (13), the updating rule for the position of each particle becomes

$$\boldsymbol{x}^{(i)}(t+1) = \boldsymbol{x}^{(i)}(t) + c_2R_2(t)(\boldsymbol{y}(t) - \boldsymbol{x}^{(i)}(t)) \quad i \in \{1,\ldots,S\} \tag{14}$$

and it is not necessary to initialize and update the velocities of the particles. Concerning the remaining parameters, we c_2 is set to 2 and the population size S is set to 25. This choice is motivated by previous analysis performed in [13], where the considered indoor environment was a cubic room with a 10 m long side, so that the volume of the room was 10^3 m^3. The simulation results in [13] indicate that a good value for the swarm size S is 200. Since the volume of the room considered in this paper to perform experiments is 75 m^3, here we set the swarm size S to 20. The stopping condition corresponds to reaching 20 iterations. The validity of this assumption will be shown at the end of Section 4.

3. Experimental setup and localization scenarios

Throughout the paper we consider localization scenarios where four RCMs are used as ANs and another RCM is considered as TN. The four RCMs are connected via USB to the same host, namely a Raspberry Pi.

The RCMs are single-board UWB radio components which can be integrated into users' electronic devices to obtain high precision range measurements. Each sensor is composed by a single board with dimensions $7.6 \times 8.0 \times 1.6\,\text{cm}^3$ and a UWB antenna. The RCMs can be programmed using C or Matlab libraries provided by Time Domain. A typical communication architecture involves many RCMs (each of which is associated with an ID) and at least one host (connected to at least one RCM via USB connection). The communication between pairs of RCMs can be carried out by directly interrogating a given ID corresponding to one of the remaining RCMs or via broadcast: in all cases, the requests are always originated by hosts. Each RCM can communicate with all the RCMs in the considered environment and each communication is associated with a message ID. The key feature of the P410 RCMs is that they provide precise and reliable range measurements, at a high update rate (up to 150 Hz), through a Two-Way ToF (TW-ToF) ranging technique. The TW-ToF technique is packet-based and involves the following steps. The requesting RCM must be connected to a host and transmits a request packet, i.e., a long (known) packet of pseudo-randomly encoded pulses. Once another RCM receives and detects this packet, it transmits a response packet to the requester, which computes the precise time delay, with accuracy on the order of picoseconds, between the request packet transmission instant and the response packet reception instant. The distance between the two communicating RCMs is then estimated by the receiving RCM, by properly processing the obtained time delay estimate, taking into account the antenna delay offset. The communication between pairs of RCMs does not only provide the estimated distance between them, but also many other data, such as: the responder ID, the relative velocity between transmitter and receiver, and an estimate of the standard deviation of the range error. After the reception of a UWB packet, the requesting RCM communicates all the data in the received packet to its host.

The main program which runs on the Raspberry Pi is written in C and is divided into two parts. In the first part, the distance estimates between each AN and the TN are acquired. All the range requests are originated from the host and all the data acquired by a sensor during a range request are communicated to the host. First, the RCMs which correspond to $\{\text{AN}_i\}_{i=2}^{M}$ are put in the sleep mode and AN_1 performs a range request by directly interrogating (through its ID) the RCM which acts as TN. As soon as the node replies, the obtained distance estimate $\hat{r}_1$ is saved and the node AN_1 is put in the sleep mode. Then, the nodes $\{\text{AN}_i\}_{i=2}^{M}$ (namely, the remaining ANs) are woken up sequentially to directly interrogate the TN. At each interrogation, the values of the estimated distances $\{\hat{r}_i\}_{i=2}^{M}$ are saved.

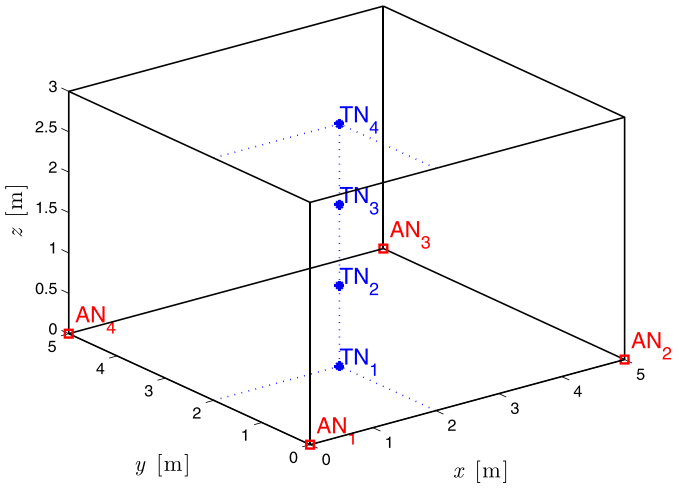

Fig. 1. First experimental configuration: the positions of the ANs (*red squares*) and the various positions of the TN (*blue dots*) are shown. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Once the range estimates $\{\hat{r}_i\}_{i=1}^{M}$ from the M ANs are acquired, in the second part of the program they are used to feed the PSO algorithm and, therefore, to localize the TN. The PSO algorithm has been implemented with the parameters described at the end of Section 2. In the following, we will also use a recently statistical model for the range estimation error [18] in order to correct the range estimates $\{\hat{r}_i\}_{i=1}^{M}$ acquired by the RCMs. As a matter of fact, the statistical model proposed in [18] is derived on the basis of an extensive experimental measurement campaign. More precisely, two RCMs have been positioned at various distances (with 1 m step) between 1 m and 15 m and, for each of these distances, 1000 range estimates have been obtained. These estimates are then used to derive a statistical model for the inter-node range estimation error. According to our results, the error which affects range estimates acquired with the RCMs can be accurately modeled as a function of the true distance between the two communicating RCMs. Denoting as r the true distance between a pair of nodes, its estimate $\hat{r}$ can be approximated as a function of r according to the following expression (dimension: [m]) [18]

$$\hat{r} \simeq r + \underbrace{0.016\,r - 0.15}_{\epsilon(r)} = 1.016\,r - 0.15 \tag{15}$$

where $\epsilon(r)$ is the range estimation error (dimension: [m]). From (15), it is clear that the range estimation error is a (linearly) increasing function of the true distance r between the two considered nodes.

According to the model in (15), the range estimates $\{\hat{r}_i\}_{i=1}^{M}$ acquired by the RCMs can be corrected as follows:

$$\check{r}_i = \frac{\hat{r}_i + 0.15}{1.016} \quad i \in \{1, \ldots, M\}. \tag{16}$$

In the following, we will compare the performance of the PSO algorithm applied to the range estimates $\{\hat{r}_i\}_{i=1}^{M}$ acquired by the RCMs with that of the same algorithm applied to the distances corrected according to (16). In [18], it is shown that the use of the model described in Eq. (16) to correct the range estimates also improves the localization accuracy of a different localization algorithm, namely the Circumference Intersection (CI) algorithm. We show that also the performance of the PSO algorithm can be improved using (16).

As anticipated at the end of Section 2, the localization scenario is a square room whose sides are 5 m long and whose height is 3 m. Different nodes configurations are considered. We first consider localization scenarios where four ANs are used to locate a TN. We consider 3 ANs' configurations and, for each of them, 4 possible TN's positions are considered: overall, this accounts for 12 different localization scenarios. The 12 TN's positions are denoted as $\{\text{TN}_i\}_{i=1}^{12}$. All these configurations are described in the following.

Configuration 1. First, we put the $M = 4$ RCMs which act as ANs on the floor, at the four corners of the room, as shown in Fig. 1. In this scenario, the ANs coordinates, expressed in meters, are:

$$\begin{aligned} &\text{AN}_1 : \boldsymbol{s}_1 = [0, 0, 0]^T \quad \text{AN}_2 : \boldsymbol{s}_2 = [5, 0, 0]^T \\ &\text{AN}_3 : \boldsymbol{s}_3 = [0, 5, 0]^T \quad \text{AN}_4 : \boldsymbol{s}_4 = [5, 5, 0]^T. \end{aligned} \tag{17}$$

Observe that with this choice of the ANs' positions, the ANs lay on the same plane. This is a typical situation in which the localization problem is ill-conditioned and classic geometric localization methods may fail, as shown in [13]. With this ANs' configuration, we consider four different TN's positions, shown in Fig. 1 as blue dots

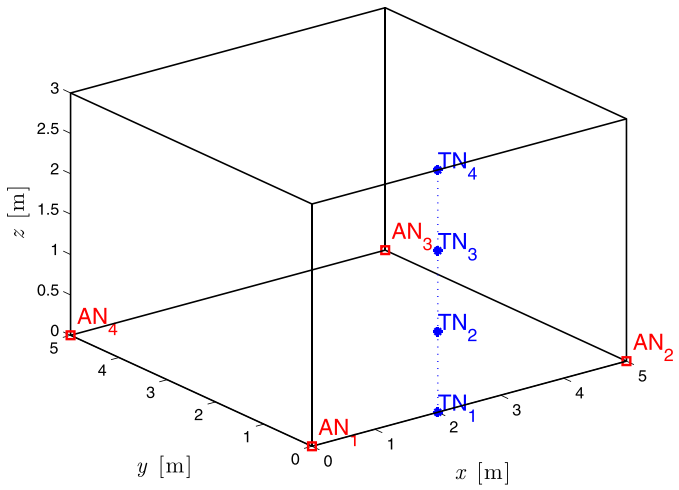

Fig. 2. Second experimental configuration: the positions of the ANs (*red squares*) and the various positions of the TN (*blue dots*) are shown. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

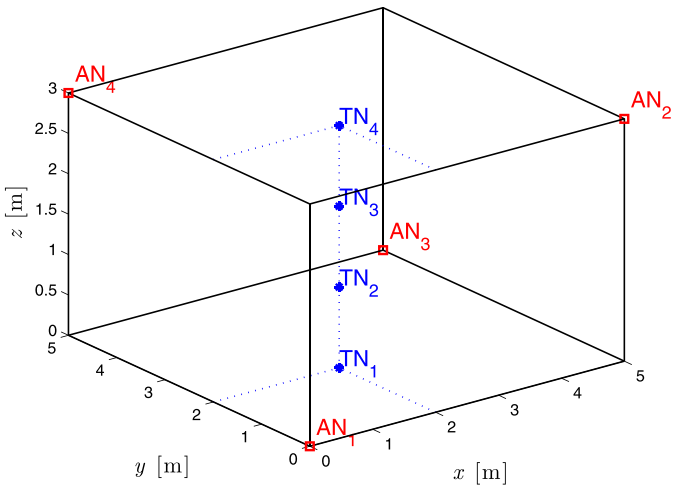

Fig. 3. Third experimental configuration: the positions of the ANs (*red squares*) and the various positions of the TN (*blue dots*) are shown. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and denoted as $\{\text{TN}_i\}_{i=1}^{4}$. The coordinates of the four TN positions, expressed in meters, are:

$$\begin{aligned} &\text{TN}_1 : \boldsymbol{u} = [2, 2, 0]^T \quad \text{TN}_2 : \boldsymbol{u} = [2, 2, 1]^T \\ &\text{TN}_3 : \boldsymbol{u} = [2, 2, 2]^T \quad \text{TN}_4 : \boldsymbol{u} = [2, 2, 3]^T. \end{aligned} \tag{18}$$

The coordinates of these four positions of the TN share the same values of abscissa and ordinate (which are both set to 2 m) but they have different heights, from 0 m (the TN lies on the floor) to 3 m (the TN lies on the ceiling). Observe that the positions of the TN are near the center of the room (with respect to the *xy* plane) at different heights.

Configuration 2. While keeping the same ANs' positions defined in (17) (i.e., in Configuration 1), we consider different positions for the TN. In particular, we put the RCM which acts a TN on a wall of the room at different heights. More precisely, we consider the following positions for the TN as shown in Fig. 2 (blue dots):

$$\begin{aligned} &\text{TN}_5 : \boldsymbol{u} = [2, 0, 0]^T \quad \text{TN}_6 : \boldsymbol{u} = [2, 0, 1]^T \\ &\text{TN}_7 : \boldsymbol{u} = [2, 0, 2]^T \quad \text{TN}_8 : \boldsymbol{u} = [2, 0, 3]^T. \end{aligned} \tag{19}$$

Configuration 3. In this case, the $M = 4$ ANs do not lie on the same plane (namely, the floor) as in the previous two configurations. The current configuration is shown in Fig. 3, where the ANs are indicated as red squares. The ANs coordinates are:

$$\begin{aligned} &\text{AN}_1 : \boldsymbol{s}_1 = [0, 0, 0]^T \quad \text{AN}_2 : \boldsymbol{s}_2 = [5, 0, 5]^T \\ &\text{AN}_3 : \boldsymbol{s}_3 = [0, 5, 0]^T \quad \text{AN}_4 : \boldsymbol{s}_4 = [5, 5, 5]^T. \end{aligned} \tag{20}$$

The positions of the TN are the same as in Fig. 1. In Fig. 3, they are denoted as $\{\text{TN}_i\}_{i=9}^{12}$ and their coordinates are:

$$\begin{aligned} &\text{TN}_9 : \boldsymbol{u} = [2, 2, 0]^T \quad \text{TN}_{10} : \boldsymbol{u} = [2, 2, 1]^T \\ &\text{TN}_{11} : \boldsymbol{u} = [2, 2, 2]^T \quad \text{TN}_{12} : \boldsymbol{u} = [2, 2, 3]^T. \end{aligned} \tag{21}$$

Configuration 4. Finally, we consider a scenario with $M = 8$ ANs. In this case, we perform localization using either (i) all the (eight) available ANs or (ii) only four of them. The localization accuracies of the two approaches will be directly compared, in order to understand the impact of the number of ANs. Unlike the previous configurations, the ANs' positions are less regular. More precisely, the ANs' coordinates, expressed in meters, are:

$$\begin{aligned} &\text{AN}_1 : \boldsymbol{s}_1 = [1, 1, 1]^T \quad \text{AN}_2 : \boldsymbol{s}_2 = [4, 0, 2]^T \\ &\text{AN}_3 : \boldsymbol{s}_3 = [3.5, 4, 1.5]^T \quad \text{AN}_4 : \boldsymbol{s}_4 = [0.5, 4.5, 2]^T \\ &\text{AN}_5 : \boldsymbol{s}_5 = [1, 2, 0]^T \quad \text{AN}_6 : \boldsymbol{s}_6 = [4.5, 3, 2.5]^T \\ &\text{AN}_7 : \boldsymbol{s}_7 = [1, 4, 0.5]^T \quad \text{AN}_8 : \boldsymbol{s}_8 = [3, 2, 3]^T. \end{aligned} \tag{22}$$

With this ANs configuration, three TN positions are considered and they are denoted as $\{\text{TN}_i\}_{i=13}^{15}$. Their coordinates are:

$$\text{TN}_{13} : \boldsymbol{u} = [3, 3, 1]^T \quad \text{TN}_{14} : \boldsymbol{u} = [1, 4, 2]^T \quad \text{TN}_{15} : \boldsymbol{u} = [5, 1, 0.5]^T. \tag{23}$$

This configuration is shown in Fig. 4 where the ANs are denoted as red squares and TNs are denoted as blue dots.

In Section 4, the performance of the PSO algorithm is analyzed in terms of (i) average localization error (i.e., average distance

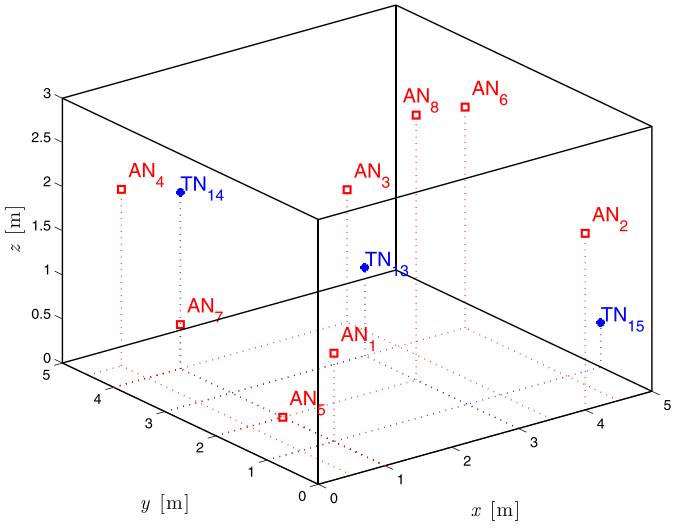

Fig. 4. Fourth experimental configuration: the positions of the ANs (*red squares*) and the various positions of the TN (*blue dots*) are shown. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

between the true and estimated TN positions) and (ii) time needed to perform PSO-based localization.

4. Experimental results

For each of the configurations described at the end of the previous section we perform $N = 100$ (independent) range estimates from the RCMs which represent the ANs and, hence, we derive 100 position estimates for each TN position. Given a TN position, let us denote as $\hat{r}_i^{(j)}$ the estimate of the distance $\hat{r}_i$ from the i-th AN in the j-th iteration, with $j \in \{1, \ldots, N\}$ and $i \in \{1, \ldots, M\}$. Moreover, we denote as $\hat{\boldsymbol{u}}^{(j)}$ the TN position estimate in the j-th iteration, obtained using the four distance estimates $\{\hat{r}_i^{(j)}\}_{i=1}^{M}$ from the four ANs.

Using the same notation introduced in Section 3, we denote as $\{\check{r}_i^{(j)}\}_{j=1}^{N}$ the distances obtained by correcting $\{\hat{r}_i^{(j)}\}_{j=1}^{N}$, $\forall i \in \{1, \ldots, M\}$, using the model introduced in (16), namely:

$$\check{r}_i^{(j)} = \frac{\hat{r}_i^{(j)} + 0.15}{1.016} \quad i \in \{1, \ldots, M\}, \quad j \in \{1, \ldots, N\}. \tag{24}$$

The TN position estimates obtained by feeding the PSO localization algorithm with the corrected distances are denoted as $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$.

In order to evaluate the performance of the proposed PSO-based localization algorithm, we consider the $N = 100$ position estimates obtained without and with the corrections on the range estimates, namely $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ and $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$, respectively.

Fig. 5 shows the position estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ (green pluses) relative to the localization of the TN denoted as TN_3 in Fig. 1. The position estimates $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ (magenta crosses) relative to the same node configuration are also shown. Such position estimates are obtained with the ANs laying on the floor, as shown in Fig. 1. It is clear from Fig. 5 that the position estimates $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ obtained using the corrective model for the range estimates (24) are closer to the true TN position that those obtained without the statistical correction. In particular, recalling that the ANs are placed on the floor (namely, they have the z-coordinate equal to 0), from Fig. 5 it can be noticed that the distance estimates obtained from the RCMs are underestimated, since all the position estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ are placed at a lower height with respect to that of the TN. The effect of the distance corrections (24) improves the localization accuracy, since the position estimates $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ are closer to the true TN position.

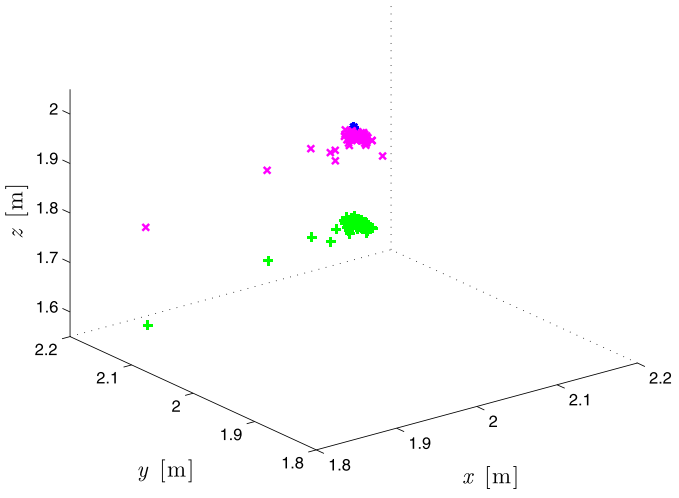

Fig. 5. The 100 position estimates $\hat{\boldsymbol{u}}^{(j)}$ (green pluses) and $\check{\boldsymbol{u}}^{(j)}$ (magenta crosses) are shown, together with the true TN position (blue dot) TN_3. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

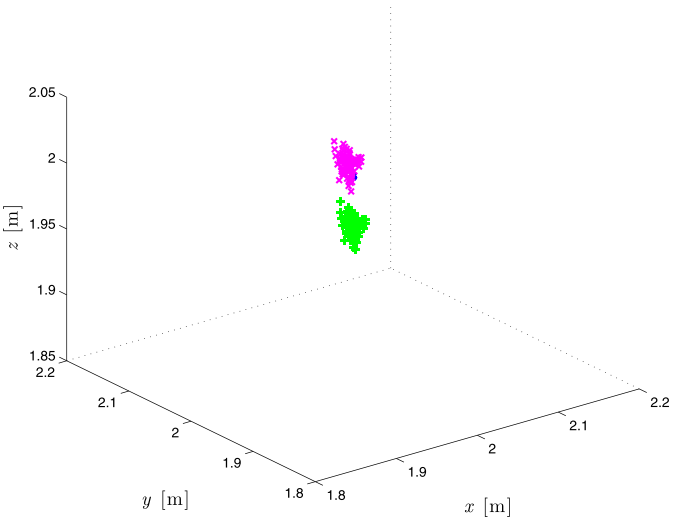

Fig. 6. The 100 position estimates $\hat{\boldsymbol{u}}^{(j)}$ (green pluses) and $\check{\boldsymbol{u}}^{(j)}$ (magenta crosses) are shown, together with the true TN position (blue dot) TN_{11}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Fig. 6, instead, shows the position estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ (green pluses) and $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ (magenta crosses) relative to the localization of the TN denoted as TN_{11} in Fig. 3. Observe that, in this scenario, the TN position is actually the same as that considered in Fig. 5. The difference with respect to the results in Fig. 5 is given by a different ANs configuration: more precisely, in Fig. 6 the ANs are placed at different heights and do not lay on the same plane. As observed for Fig. 5, also in Fig. 6 the position estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ obtained using the statistical correction for the range estimates (24) are closer to the true TN position that $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$, showing the validity of the proposed model. Comparing the position estimates in Fig. 5 with those in Fig. 6, it can be observed that, considering the same TN position, the ANs configuration of Fig. 3 allows obtaining better performance with respect to that of Fig. 1. The PSO-based localization algorithm, however, guarantees a sufficiently accurate localization also in the least favorable case where the ANs lay on the same plane, while more classic localization methods would fail. As a matter of fact, classic localization algorithms typically rely on the solution of linear or non-linear systems of equations, which can become ill-conditioned in correspondence to specific nodes' topologies. In particular, if the ANs lay on the same line or plane, the position estimate in a three-dimensional environment cannot be evaluated using "geometrical" algorithms, such as the TSML algorithm [8] or the Plane Intersection (PI) algorithm [6], which are classic localization methods. At the opposite, the approach based on the PSO algorithm allows accurate position estimation, regardless of the configuration of nodes (both ANs and TN). A direct simulation-based performance comparison between a PSO-based localization approach and classic geometric localization algorithms can be found in [13,19].

We now consider a scenario where the TN is placed at a smaller height. In particular, Fig. 7 shows the position estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ (green pluses) and $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ (magenta crosses) relative to the localization of the TN denoted as TN_2 in Fig. 1. Such position estimates are obtained in the scenario shown in Fig. 1, namely with the ANs laying on the floor. The TN position estimates $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ in Fig. 7 are once again closer to the true TN position than those obtained

Table 1
The values of $\hat{d}_{\text{avg}}$ and $\check{d}_{\text{avg}}$ relative to the 12 TN positions associated with Configurations 1, 2, and 3 are shown.

	$\hat{d}_{\text{avg}}$ [cm]	$\check{d}_{\text{avg}}$ [cm]		$\hat{d}_{\text{avg}}$ [cm]	$\check{d}_{\text{avg}}$ [cm]		$\hat{d}_{\text{avg}}$ [cm]	$\check{d}_{\text{avg}}$ [cm]
TN_1	8.2	4.7	TN_5	53.5	39.9	TN_9	20.5	14.7
TN_2	53.5	4.8	TN_6	37.6	28.8	TN_{10}	10.8	4.2
TN_3	21.7	4.4	TN_7	40.6	32.2	TN_{11}	7.9	2.8
TN_4	40.6	35.4	TN_8	56.4	46.3	TN_{12}	29.3	22.7

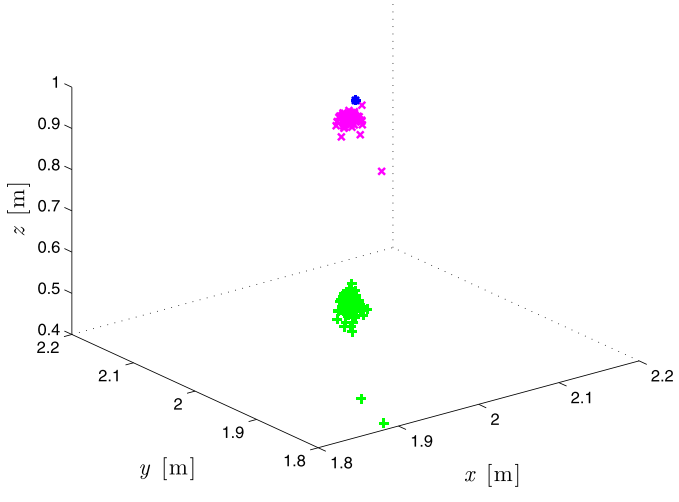

Fig. 7. The 100 position estimates $\hat{\boldsymbol{u}}^{(j)}$ (green pluses) and $\check{\boldsymbol{u}}^{(j)}$ (magenta crosses) are shown, together with the true TN position (blue dot) TN_2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

without the range estimate model. As a matter of fact, the height of the position estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ is far smaller (by nearly half a meter) than the true TN height. Observe that the improvement due to the use of the model (24) is more evident in this case than in Fig. 5.

Finally, we consider the position estimates relative to the localization of the TN in the same position of Fig. 7, but with a different ANs' configuration. More precisely, Fig. 8 shows the position estimates relative to the TN placed in the position denoted as TN_{10} in Fig. 3, where the ANs do not lay on the same plane. The results in Fig. 8 show that, also in this case, the position estimates $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$

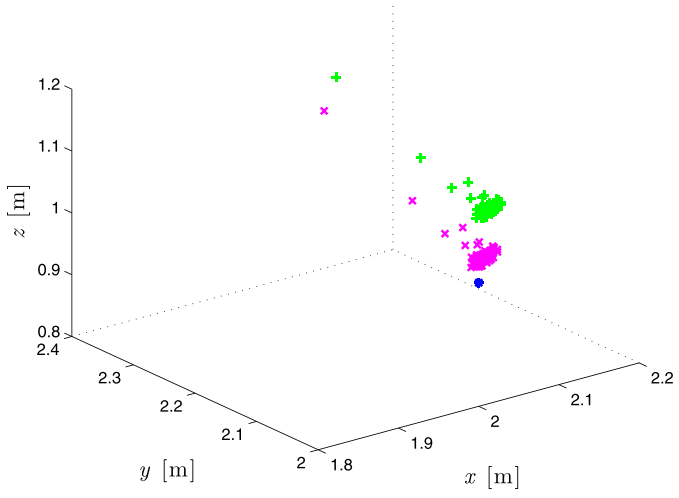

Fig. 8. The 100 position estimates $\hat{\boldsymbol{u}}^{(j)}$ (green pluses) and $\check{\boldsymbol{u}}^{(j)}$ (magenta crosses) are shown, together with the true TN position (blue dot) TN_{10}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

(magenta crosses) obtained using the range estimates $\{\check{r}_i^{(j)}\}_{j=1}^{N}$ are closer to the true TN position than $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ (green plus).

In order to concisely describe the accuracy of the proposed localization strategy, it is expedient to evaluate the average distance between the true TN position $\boldsymbol{u}$ and its estimates averaged over the N position estimates. We consider the position estimates $\hat{\boldsymbol{u}}^{(j)}$ obtained using the range estimates acquired by the nodes and also the position estimates $\hat{\boldsymbol{u}}^{(j)}$ obtained by applying the PSO-based localization algorithm to the distance estimates $\{\check{r}_i^{(j)}\}_{j=1}^{N}$ corrected according to (24). Let us define as

$$\hat{d}^{(j)} \triangleq ||\boldsymbol{u} - \hat{\boldsymbol{u}}^{(j)}|| \tag{25}$$

the distance between the true TN position $\boldsymbol{u}$ and its estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ in the j-th iteration. Similarly, we define as

$$\check{d}^{(j)} \triangleq ||\boldsymbol{u} - \check{\boldsymbol{u}}^{(j)}|| \tag{26}$$

the distance between the true TN position $\boldsymbol{u}$ and its estimates $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ in the j-th iteration. The average distances $\hat{d}_{\text{avg}}$ and $\check{d}_{\text{avg}}$ are then defined as

$$\hat{d}_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N}\hat{d}^{(j)} \quad \check{d}_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N}\check{d}^{(j)}. \tag{27}$$

The values of $\hat{d}_{\text{avg}}$ and $\check{d}_{\text{avg}}$ relative to the 12 TN positions and based to our experimental results are shown in Table 1. For every TN position the values of $\check{d}_{\text{avg}}$ are lower than those of $\hat{d}_{\text{avg}}$, meaning that the TN position estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ obtained without the range model (24) are, on average, less accurate than $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ which are obtained using the range model. The difference between $\hat{d}_{\text{avg}}$ and $\check{d}_{\text{avg}}$ is particularly evident when considering the TN position TN_2. In this case, $\check{d}_{\text{avg}}$ is lower than 10% of $\hat{d}_{\text{avg}}$. Table 1 also shows that the localization error is larger when the TN is near a wall (namely, when considering $\{TN_i\}_{i=5}^{8}$) rather than when the TN is near the center of the room.

As previously observed, the ANs' topologies in *Configuration 1* and *Configuration 2* do not allow locating the TN when using classic geometric localization approaches, such as the TSML algorithm, since the ANs lay on the same plane and the underlying mathematical problem becomes ill-conditioned. In *Configuration 3*, instead, the matrix involved in the TSML algorithm is not ill-conditioned and, therefore, the TSML algorithm does not fail. In order to compare the performance of the PSO-based localization algorithm with that of the TSML localization algorithm, we apply the latter to locate $\{TN_i\}_{i=9}^{12}$ with the ANs' topology shown in Fig. 3. The values of $\hat{d}_{\text{avg}}$ and $\check{d}_{\text{avg}}$ obtained in this case are shown in Table 2. Comparing the values of $\check{d}_{\text{avg}}$ relative to the TSML algorithm in Table 2 with those relative to the PSO algorithm in Table 1, it can be observed that the TSML algorithm is more accurate than the PSO-based algorithm when locating TN_9, while it is less accurate when locating TN_{10}. In the last two rows of Table 2, namely when considering TN_{11} and TN_{12}, the performance of the two algorithms is similar. The obtained results show that, when the nodes' topology makes the TSML algorithm applicable, its performance is similar to that of

Table 2
The values of $\hat{d}_{avg}$ and $\check{d}_{avg}$ relative to the 4 TN positions shown in Fig. 3 obtained with the TSML localization algorithm.

	$\hat{d}_{avg}$ [cm]	$\check{d}_{avg}$ [cm]
TN_9	4.3	4.2
TN_{10}	33.5	33.1
TN_{11}	4.5	3.7
TN_{12}	21.6	20.1

Table 3
The values of $\hat{d}_{avg}$ and $\check{d}_{avg}$ relative to the 3 TN positions shown in Fig. 4 (namely: TN_{13}, TN_{14}, and TN_{15}) obtained with 8 ANs or 4 ANs, respectively, are shown.

	$\hat{d}_{avg}$ [cm] (8 ANs)	$\hat{d}_{avg}$ [cm] (4 ANs)	$\check{d}_{avg}$ [cm] (8 ANs)	$\check{d}_{avg}$ [cm] (4 ANs)
TN_{13}	16.5	16.8	3.5	2.8
TN_{14}	21.6	15.3	14.1	10.8
TN_{15}	24.9	15.6	21.7	14.3

the PSO-based approach. However, the latter approach is applicable in every scenario and does not suffer of ill-conditioning.

Finally, we consider the nodes configuration in Fig. 4 where eight ANs are available to perform localization. Fig. 9 shows the position estimates obtained by applying the PSO-based approach described in Section 2 to localize the TN denoted as TN_{14} in Fig. 4. More precisely, the 100 position estimates obtained by applying the PSO-based localization algorithm described in Section 2 with 8 ANs are shown with dark green triangles. Green pluses, instead, show the 100 position estimates $\{\hat{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ obtained by applying the PSO-based localization algorithm with the 4 ANs nearest to the TN position, namely: AN_3, AN_4, AN_5, and AN_7 in Fig. 4. We remark that four is the minimum number of ANs which are necessary to perform localization. The results in Fig. 9 show that performing localization using only the four ANs nearest to the TN (instead of all the available ones) allows a more accurate TN localization. Similar results are obtained when applying the statistical correction for the range estimates given by (24). In particular, Fig. 9 also shows the corresponding 100 position estimates $\{\check{\boldsymbol{u}}^{(j)}\}_{j=1}^{N}$ obtained by applying the PSO-based localization algorithm with 8 ANs (violet diamonds) and with 4 ANs (magenta crosses). From Fig. 9, it can be concluded that also when using the statistical correction (24) the TN's position estimates are more accurate if only the four ANs nearest to the TN are used to perform localization.

In Table 3, the values of $\hat{d}_{avg}$ and $\check{d}_{avg}$ (defined in (27)) are shown for the TN positions $\{TN_i\}_{i=13}^{15}$ shown in Fig. 4. More precisely, Table 3 shows the values of $\hat{d}_{avg}$ and $\check{d}_{avg}$ obtained when performing localization using the range measurements from 8 ANs and when using only the range estimates from the 4 ANs which are nearest to the TN. In all cases, as predicted by the results in Fig. 9, the PSO-based localization accuracy is highest by relying only on the range measurements from the 4 ANs which are nearest to the TN. These results are reasonable since, according to (15), the range estimation error between two nodes is an increasing function of the distance between the nodes. Therefore, the position estimates obtained relying only on the 4 ANs nearest to the TN are, on average, more accurate than those obtained when using all the available ANs. As a matter of fact, according to (15), less accurate range estimates from far ANs have a negative impact on the final TN's position estimate.

We now show, through some illustrative examples, the validity of the assumption, introduced at the end of Section 2, according to which 20 iterations of the PSO algorithm are sufficient to perform localization. In Fig. 10 the distance between the true TN position and the particle of the swarm with the best position (in the first of the $N = 100$ execution of the localization process) is shown as a function of the number of iterations $n \in \{0, \ldots, 20\}$, considering TN_1 (Fig. 10 (a.)) and TN_9 (Fig. 10 (b.)) as positions for the TN. In, Fig. 10, the solid green line is associated with the distance between the true TN position (TN_1 in Fig. 1) and the particle of the swarm with the best position when using $\{\hat{r}_i^{(1)}\}_{i=1}^{4}$. In the same figure, the dashed magenta line is instead associated with the distance between the true TN position and the particle of the swarm with the best position when using the corrected distances $\{\check{r}_i^{(1)}\}_{i=1}^{4}$. The results in Fig. 10 (b.) relative to the case with the TN' position TN_9 shown in Fig. 3 is analogous to Fig. 10 (a.), but is relative to the TN position TN_9 in Fig. 3 (where the ANs are placed at different heights) are shown. In each of the two cases, the localization error converges to its minimum before reaching 20 iterations.

Fig. 11 shows the distance between the true TN position and the particle of the swarm with the best position as a function of the number of iterations, considering the TN in the positions (a.)

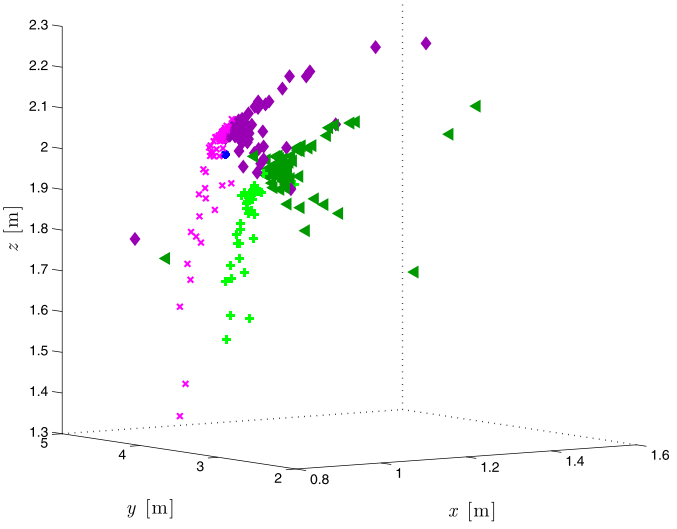

Fig. 9. The 100 position estimates $\hat{\boldsymbol{u}}^{(j)}$ obtained when using 8 ANs (dark green triangles) and when using 4 ANs (green pluses) are shown. The 100 position estimates $\check{\boldsymbol{u}}^{(j)}$ obtained when using 8 ANs (violet diamonds) and when using 4 ANs (magenta crosses) are also shown, together with the true TN position (blue dot) TN_{14}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

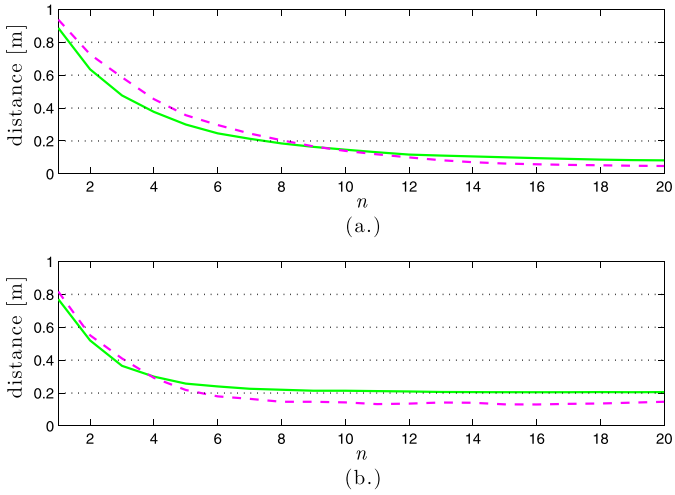

Fig. 10. The distance between the true TN position and the particle of the swarm with the best position when using $\{\hat{r}_i^{(1)}\}_{i=1}^{4}$ (solid green line) and when using $\{\check{r}_i^{(1)}\}_{i=1}^{4}$ (dashed magenta line) is shown when considering the TN positions: (a.) TN_1 and (b.) TN_9. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

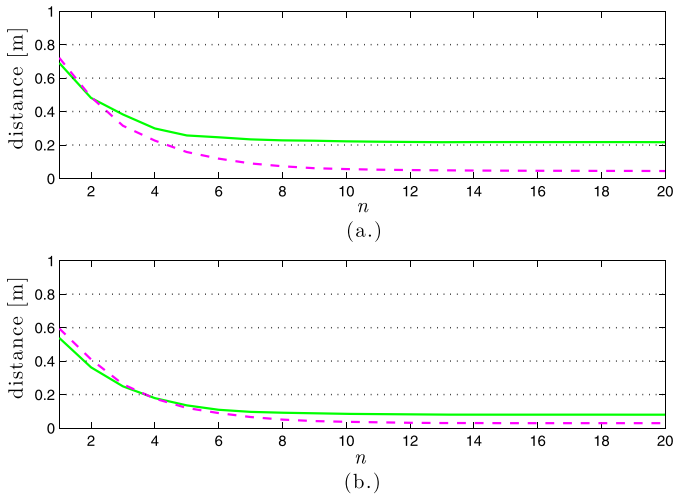

Fig. 11. The distance between the true TN position and the particle of the swarm with the best position when using $\{\hat{r}_i^{(1)}\}_{i=1}^{4}$ (solid green line) and when using $\{\check{r}_i^{(1)}\}_{i=1}^{4}$ (dashed magenta line) is shown when considering the TN positions: (a.) TN_3 and (b.) TN_{11}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

TN_3 (illustrated in Fig. 1) and (b.) TN_{11} (illustrated in Fig. 3). The solid green lines are relative to the results obtained using the distance estimates $\{\hat{r}_i^{(1)}\}_{i=1}^{4}$, while the dashed magenta lines refer to the results obtained with $\{\check{r}_i^{(1)}\}_{i=1}^{4}$.

Finally, let us make a few considerations on the execution time. The acquisition of the distance estimate from each of the four RCMs acting as ANs requires 8 ms on the Raspberry Pi. The execution of the 20 PSO iterations with a swarm size equal to 25 requires 0.1 ms. Therefore, a single localization estimate can be performed in 32.1 ms so that at least 30 TN position estimates per second can be performed. The potentially high update rate makes the proposed PSO-based localization algorithm attractive for dynamic localization and, ultimately, leads naturally to its application to tracking.

5. Conclusions

In this paper, a PSO-based localization algorithm is derived and applied in a realistic three-dimensional context. More precisely, we have considered an experimental testbed composed by a set of five PulsOn 410 RCMs produced by Time Domain. Such nodes rely on the UWB technology to estimate inter-node distances. Given the range estimates between four or eight ANs and the TN, the position of the latter in a 3D environment is estimated according to the proposed PSO-based localization algorithm. Our experimental results show that the performance of the proposed algorithm is good regardless of the ANs topology. In particular, localization is carried out successfully also in scenarios where classic geometric approaches (e.g., TSML) fail.

Moreover, a recently proposed statistical correction of the range estimation error allows to further improve the performance of the proposed localization strategy. Finally, the execution time of the proposed PSO-based approach can be kept small, which makes it suitable also for dynamic localization.

References

[1] S. Monica, G. Ferrari, Accurate indoor localization with UWB wireless sensor networks, in: Proceedings of the 23rd IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises *(WETICE 2014)*, track on Capacity-Driven Processes and Services for the Cyber-Physical Society *(CPS)*, Parma, Italy, 2014, pp. 287–289.
[2] J. Zhang, P.V. Orlik, Z. Sahinoglu, A.F. Molisch, P. Kinney, UWB systems for wireless sensor networks, Proc. IEEE 97 (2) (2009) 313–331.
[3] S. Gezici, H.V. Poor, Position estimation via Ultra-Wide-Band signals, Proc. IEEE 97 (2) (2009) 386–403.
[4] W.H. Foy, Position-location solutions by Taylor-series estimation, IEEE Trans. Aerosp. Electron. Syst. 12 (2) (1976) 187–194.
[5] C. Mensing, S. Plass, Positioning algorithms for cellular networks using TDOA, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing *(ICASSP 2006)*, Toulouse, France, 2006, pp. 513–516.
[6] R.O. Schmidt, A new approach to geometry of range difference location, IEEE Trans. Aerosp. Electron. Syst. 8 (6) (1972) 821–835.
[7] G. Shen, R. Zetik, R.S. Thomä, Performance comparison of TOA and TDOA based location estimation algorithms in LOS environment, in: Proceedings of the 5th Workshop on Positioning, Navigation and Communication *(WPNC'08)*, Hannover, Germany, 2008, pp. 71–78.
[8] Y. Chan, K.C. Ho, A simple and efficient estimator for hyperbolic location, IEEE Trans. Signal Process. 42 (8) (1994) 1905–1915.
[9] S. Thrun, D. Fox, W. Burgard, F. Dellaert, Robust Monte Carlo localization for mobile robots, Artif. Intell. 128 (1) (2001) 99–141.
[10] Y. Senta, Y. Kimuro, S. Takarabe, T. Hasegawa, Machine learning approach to self-localization of mobile robots using RFID tag, in: IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2007, pp. 1–6.
[11] E. Okamoto, M. Horiba, K. Nakashima, T. Shinohara, K. Matsumura, Particle swarm optimization-based low-complexity three-dimensional UWB localization scheme, in: Proceedings of the International Conference on Ubiquitous and Future Networks, 2014, pp. 120–124.
[12] S. Monica, G. Ferrari, Swarm intelligent approaches to auto-localization of nodes in static UWB networks, Appl. Soft Comput. 25 (2014) 426–434.
[13] S. Monica, G. Ferrari, A swarm intelligence approach to 3D distance-based indoor UWB localization, in: Proceedings of the International Conference on the Applications of Evolutionary Computation *(EvoApplications '15)*, track on Nature-inspired Techniques for Communication Networks and other Parallel and Distributed Systems *(EvoCOMNET '15)*, Copenaghen, Denmark, 2015.
[14] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks *(ICNN)*, Perth, Australia, 1995, pp. 1942–1948.
[15] R. Poli, J. Kennedy, T. Blackwell, Particle swarm optimization, Swarm Intell. J. 1 (1) (2007) 33–57.
[16] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: Proceedings of the IEEE International Conference on Evolutionary Computation *(ICEC)*, Washington, DC, 1999, pp. 69–73.
[17] X.S. Yang, S. Deb, S. Fong, Accelerated particle swarm optimization and support vector machine for business optimization and applications, Commun. Comput. Inf. Sci. 136 (2011) 53–66.
[18] S. Monica, G. Ferrari, An experimental model for UWB distance measurements and its application to localization problems, in: Proceedings of the IEEE International Conference on Ultra Wide Band *(ICUWB 2014)*, Paris, France, 2014, pp. 297–302.
[19] S. Monica, G. Ferrari, Particle swarm optimization for auto-localization of nodes in wireless sensor networks, in: Proceedings of the 11th International Conference on Adaptive and Natural Computing Algorithms *(ICANNGA '13)*, Lausanne, Switzerland, 2013, pp. 456–465.
[20] S. Monica, G. Ferrari, Optimized anchors placement: an analytical approach in UWB-based TDOA localization, in: Proceedings of the 9th International Wireless Communications & Mobile Computing Conference *(IWCMC 2013)*, Cagliari, Italy, July, 2013, pp. 982–987.