Experimental Results on Agent-Based Indoor Localization using WiFi Signaling

Stefania Monica, Federico Bergenti
Dipartimento di Scienze Matematiche, Fisiche e Informatiche
Università degli Studi di Parma
Parco Area delle Scienze 53/A, 43124 Parma, Italy

***Abstract*—This paper discusses experimental results on the possibility of accurately estimating the position of smart devices in known indoor environments using agent technology. Discussed localization approaches are based on WiFi signaling, which can be considered as an ubiquitous technology in the large majority of indoor environments. The use of WiFi signaling ensures that no specific infrastructures nor special on-board sensors are required to support localization. Localization is performed using range estimates from the fixed access points of the WiFi network, which are assumed to have known positions. The performance of two range-based localization algorithms are discussed. The first, called Two-Stage Maximum-Likelihood algorithm, is well-known in the literature, while the second is a recent optimization-based algorithm that uses particle swarm techniques. Results discussed in the last part of the paper show that a proper processing of WiFi-based range estimates allows obtaining accurate position estimates, especially if the optimization-based algorithm is used.**

Keywords*—*WiFi-based localization; indoor localization; particle swarm optimization; agent technology

I. Introduction

In recent years, mobile devices and smart appliances have assumed a relevant impact on everyday life and their number has rapidly increased. Such devices can be commonly used in a variety of contexts, including, e.g., fitness activity and sport, tourism and outdoor navigation, monitoring of environmental parameters, social networks and social games. One of the weaknesses observed in available smart appliances is that they do not yet offer support for accurate indoor localization. While outdoor localization is effectively achieved using various assisted technologies, such as the *Global Positioning System (GPS)*, accurate indoor localization is still an open issue (e.g., [1], [2]). Recently, the use of specific technologies to support indoor localization has been investigated. Among such technologies, it is worth recalling *Ultra Wide Band (UWB)*, which seems to be very promising (e.g., [3], [4]). The large bandwidth and the high time resolution of UWB signals reduce the impact of phenomena which typically interfere with wireless communications in indoor environments (e.g., non-line-of-sight propagation, multi-path, and multiple access interference) [5]. The main drawback of this technology is that it requires a dedicated infrastructure which is not widely available in common indoor scenarios, and which is also still expensive today. In order to overcome such a limitation, we focus in this paper on indoor localization approaches based on the use of WiFi technology, which can be easily found in almost all indoor environments.

In the experimental scenario discussed in this paper, WiFi-based localization is addressed by means of agent technology. Presented experimental results are obtained by using an add-on module for the *Java Agent and DEvelopment framework* (*JADE*) [6]. JADE has been developed during the last 20 years and, in 1998, four of the major manufactures of mobile appliances of the time started a joint research initiative [7] to bring agent technology to what we used to call *Java-enabled phones* at the time. The results of such an initiative eventually became the base of JADE for Android [8]. Since then, *nomadic agents* have been considered one of the most promising applications of agent technology and JADE, together with its companion language JADEL [9]–[11], is a consolidated tool in this field which has been used for many applications (e.g., [12]–[14]). The significant opportunities that the synergic combination of agents and smart appliances offer have already been investigated, e.g., in the scope of the *Agent-based Multi-User Social Environment* (*AMUSE*) project [15]. AMUSE is a major evolution of JADE and it consists of an open-source platform built on top of JADE that addresses specific issues of online social games, which is an application domain where the use of agents is particularly promising. AMUSE has already been fruitfully used to experiment mixed-reality games, in which the possibility of interacting with the physical world becomes crucial [16]. In addition, other examples of games that can be developed using AMUSE include social games in indoor areas with high concentration of potential users, like halls of shopping malls, waiting areas of airports and train stations, interactive museums and exhibits, and covered markets in historic towns. Such areas typically offer dedicated WiFi coverage by means of *Access Points* (*APs*) spread in the environment, which can be used to support effective localization by means of dedicated techniques as discussed in the following sections. Note that discussed techniques only requires that the WiFi receiver is active on smart appliances and they do not necessitate that appliances are connected to one of the WiFi networks of the area.

The JADE module used to obtain experimental results discussed in this paper can be used to develop agents capable of sensing their positions with respect to a fixed reference frame in known indoor environments [17]. Localization is performed by properly processing estimates of the distances between the smart appliance where the agent runs and the APs of the WiFi network, whose positions are assumed to be known. In such scenarios, estimated distances among the smart appliance and APs can be used to feed a *localization algorithm* which is in charge of providing the agent with an estimate of its position. Experimental results discussed in this paper show that the accuracy of the two discussed localization algorithms are comparable. Actually, both can achieve an accuracy in the order of 1 m, and even better accuracy can be obtained by properly averaging range estimates. Note that the paper focuses on presenting two of the localization algorithms available in the JADE add-on module, and a detailed description of the architecture of the module is left for a future paper.

This paper is organized as follows. Section II introduces the problem of localization and discusses relevant related work. Section III shows how WiFi signaling can be used to obtain information on the position of smart appliances and introduces useful notations. Section IV illustrates the two discussed localization algorithms. Section V introduces metrics to evaluate the performance of discussed approaches and shows illustrative experimental results. Section VI concludes the paper.

II. Relevant Approaches for Indoor Localization

Many localization algorithms have been proposed in the literature to provide location information to nodes of a network (e.g., [18]). In particular, *range-based localization methods* rely on the knowledge of inter-node distances or angle information and they can be classified into *active* and *passive* [3]. In active methods, all nodes are equipped with sensors and with an electronic device which sends information to a positioning system. Passive localization, instead, is based on the fact that wireless communications strongly depend on the surrounding environment. Relying on the scattering caused by small targets during signal propagation and/or on the variance of a measured signals, changes in the received signals can be used to detect and locate targets and for tracking purposes [19].

In this paper, we focus on active range-based algorithms, which typically involves two steps. First, proper parameters related to signals traveling between the smart appliance and some nodes of the network with known positions are estimated. Such parameters can be the *Time of Flight* (*ToF*), the *Angle of Arrival* (*AoA*), or the *Received Signal Strength* (*RSS*) [20]. Then, the parameters evaluated in the first step are used to estimate the position of the smart appliance using a proper localization algorithm. Concerning the first step, note that range-estimation techniques based on AoA rely on the measurements of angles between nodes, which are usually taken by means of antenna arrays that require dedicated hardware. Moreover, the installation cost of antenna arrays can be high and the number of signal paths in indoor environments can be large for the presence of obstacles, which make accurate angle estimation challenging in practical scenarios. Time-based range-estimation techniques, instead, rely on measurements of the ToF of signals traveling between nodes, and, therefore, they require high time resolution in the processing of considered signals and they are not particularly well suited when WiFi technology is used. For the reasons mentioned above, we focus on localization approaches based on RSS, which is typically used when performing localization with WiFi signaling. Under the assumption that the energy of transmitted signals is known, the availability of a relation between the received power of a signal traveling between two nodes and the distance between the nodes can be used to estimate the distance using RSS measurements at the receiver node. In detail, in order to use RSS measurements to estimate the distance that separates a receiving node from a transmitting node, we rely on the *Friis transmission equation*, according to which the received power $P(\rho)$ at distance ρ can be expressed as [5]

$$P(\rho) = P_0 - 10\beta \log_{10} \frac{\rho}{\rho_0}, \tag{1}$$

where, P_0 is the known power at reference distance ρ_0 and β is a parameter related to the details of the transmission [21]. An estimate of the received power $P(\rho)$ yields the value of the distance ρ by inverting (1).

In the remaining of this paper, we assume that range estimates are obtained from measurements of the RSS of signals traveling between nodes using (1). The adopted add-on module for JADE uses the functionality of the smart appliance where it is installed to obtain the RSS of signals used by WiFi APs to support network scanning. Measured RSS is used to estimate the distances between the smart appliance where the module is installed, which is normally called *Target Node* (*TN*), and the APs of the network, which are assumed to be in known and static positions. Each communication between the TN and one of the AP allows obtaining an estimate of the distance between them together with other valuable information, such as the *Basic Service Set IDentification* (*BSSID*) of the responding AP. Assuming that the position of each AP is known, each mapped BSSID can be associated with the coordinates of the corresponding AP and, hence, each distance estimate can also be related to the coordinates of the corresponding AP. The possibility of associating the position of an AP with each distance estimate between a TN and that AP is crucial to support localization, as discussed in the following.

In order to provide agents with information on their current positions, the add-on module for JADE integrates localization algorithms that use estimated distances from APs to estimate the position of the TN and feed it to interested agents. In detail, once range estimates are acquired, they are used to feed one of the available localization algorithms, which computes an estimate of the position of the TN. Finally, interested agents are informed of the current position of the TN, which is used as the position of the agents that are host in the smart appliance where the add-on module is installed. The current version of the JADE module includes different localization algorithm and, in this paper, we compare the performance of two of them:

- The *Two-Stage Maximum-Likelihood* (*TSML*) algorithm, which is well-known in the literature because it can attain the Cramer-Rao lower bound, which is a lower bound for the variance of an estimator [22].
- A recent optimization-based algorithm [23], which relies on the use of *Particle Swarm Optimization* (*PSO*) to obtain effective localization.

Preliminary results on the accuracy of discussed algorithms were presented in [24], where only the TSML algorithm was considered to address a localization scenario different from the scenario discussed in the following sections. Experimental results discussed in this paper show that the accuracy of the two localization algorithms are comparable, and that they can both achieve an accuracy in the order of 1 m. Such accuracy can be further improved, especially for the second algorithm, by using proper processing of estimated distances.

III. WiFi-Based Range Estimates

In this section, we introduce relevant notation that will be used to describe the localization scenario and localization algorithms. Throughout the paper, we denote the number of available APs as M. Note that the coordinates of APs are assumed to be known and static, and using a proper coordinate system, they are denoted as

$$\underline{s}_i = (x_i, y_i, z_i)^{\mathsf{T}}. \tag{2}$$

The unknown position of the TN is denoted as

$$\underline{u} = (\bar{x}, \bar{y}, \bar{z})^{\mathsf{T}}. \tag{3}$$

For the sake of simplicity, in the following we assume that there exist a unique TN whose position needs to be estimated, but it is worth noting that discussed approaches can be used also with a large number of TN, provided they are all equipped with the JADE add-on module. Using the introduced notation, the distance between the TN and the i-th AP is

$$\rho_i = ||\underline{u} - \underline{s}_i||,$$

where $||\underline{x}||$ denotes the Euclidean norm of vector $\underline{x}$. The position of the TN defined in (3) can be found by intersecting the spheres with radii $\{\rho_i\}_{i=1}^{M}$ centered in $\{\underline{s}_i\}_{i=1}^{M}$, i.e., by solving the following non-linear system of equations:

$$\begin{cases} (x-x_1)^2 + (y-y_1)^2 + (z-z_1)^2 = \rho_1^2 \\ \vdots \\ (x-x_M)^2 + (y-y_M)^2 + (z-z_M)^2 = \rho_M^2. \end{cases} \tag{4}$$

In order to guarantee that the system of equations (4) has a unique solution, the number of available APs needs to be at least equal to 4.

Unfortunately, when performing localization in realistic scenarios, the exact values of distances $\{\rho_i\}_{i=1}^{M}$ between the TN and APs are unknown. As a matter of fact, their knowledge together with the system of equations (4) would also imply that the true TN position $\underline{u}$ would be known. For this reason, it is necessary to rely on range estimates acquired by a proper processing of the RSS of WiFi signals. As soon as range estimates from the M APs are available, a proper localization algorithm can use them to estimate the position of the TN. It is assumed that this acquisition and processing procedure is iterated $L > 1$ times, thus leading to L position estimates at different instants. We denote the estimated distance between the $i-$th AP and the TN at the $j-$th iteration as $\hat{\rho}_{i,j}$.

Before describing the localization algorithms used to obtain discussed results, let us make additional comments on range estimates. Instead of using a single range estimate from each APs, it is possible to acquire $K > 1$ range estimates from each AP and then use the average of such range estimates to derive an estimate of the TN position. In detail, let us define

$$\hat{\rho}_{i,j}^{K} = \frac{1}{K} \sum_{h=j}^{j+K-1} \hat{\rho}_{i,h} \tag{5}$$

which represent the averaged range estimates from the i-th AP obtained by averaging $\hat{\rho}_{i,j}$ over K consecutive acquisitions. Averages $\hat{\rho}_{i,j}^{K}$ can be used instead of single estimates $\hat{\rho}_{i,j}$ to alleviate problems related to acquisition noise and multipath.

From now on, we make an additional assumption which allows simplifying the localization algorithm, i.e., we assume that the height $\bar{z}$ of the considered TN is known. Even though this may seem a strong assumption, we remark that, in considered scenarios, users are holding their smart appliances, i.e., the TNs, in their hands or in their pockets. Hence, even if the true height is not accurately known, it can be reasonably approximated to, e.g., $\bar{z} = 1$ m. Errors in the order of a few centimeters on the value of $\bar{z}$ do not have a strong impact on the accuracy of discussed algorithms because range estimates error are typically in the order of 10 cm [25]. Additionally, this assumption has the advantage of simplifying the localization algorithm as if the considered scenario was a bidimensional one, namely, as if the coordinates of the i-th AP were $(x_i, y_i, \bar{z})$ for $i \in \{1, \ldots, M\}$. In other words, the third coordinate can be neglected and the mathematical description of the discussed localization algorithms is simplified. Let us define the difference between the height $\bar{z}$ of the TN and the height z_i of the i-th AP as

$$h_i = ||\bar{z} - z_i||. \tag{6}$$

Given this definition, it is possible to evaluate the projections of the distances $\{\rho_i\}_{i=1}^{M}$ between the TN and the i-th AP on the plane $z = \bar{z}$ where the TN lies. According to the Pythagorean theorem, under the assumption that the true values of distances $\{\rho_i\}_{i=1}^{M}$ are known, the projections of such distances can be written as

$$r_i = \sqrt{\rho_i^2 - h_i^2}. \tag{7}$$

An illustrative geometric representation of the relations among ρ_i, h_i, and r_i is shown in Fig. 1, assuming that the coordinates of the considered AP, expressed in meters, are $(0, 0, 3)$ and those of the TN, expressed in meters, are $(1, 1, 1)$.

Under the assumption that that the height $\bar{z}$ of the TN is known, the abscissa and the ordinate of the TN can be found by intersecting the circumferences centered in $\{(x_i, y_i)\}_{i=1}^{M}$ with radii $\{r_i\}_{i=1}^{M}$, i.e., by solving the following non-linear system of equations:

$$\begin{cases} (x-x_1)^2 + (y-y_1)^2 = r_1^2 \\ \vdots \\ (x-x_M)^2 + (y-y_M)^2 = r_M^2. \end{cases} \tag{8}$$

As previously observed, the values of $\{\rho_i\}_{i=1}^{M}$ are unknown and, consequently, the values of $\{r_i\}_{i=1}^{M}$ are also unknown. Hence, it is necessary to rely on range estimates obtained from the RSS of WiFi signals. Let us define the projections of distances $\hat{\rho}_{i,j}$ from the $i-$th AP at the $j-$th iteration on the plane $z = \bar{z}$ as

$$\hat{r}_{i,j} = \sqrt{\hat{\rho}_{i,j}^2 - h_i^2}. \tag{9}$$

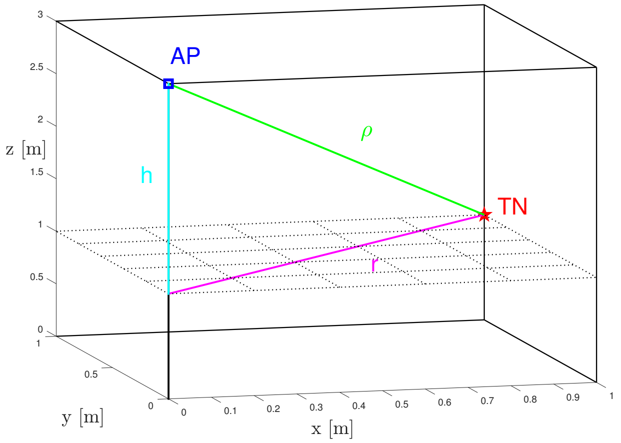

Fig. 1. The distance ρ between an AP (blue square) and a TN (red star) is shown (green line), together with the height h of AP with respect to the TN (cyan line). The projection r of the distance ρ on the plane on which the TN lies is also shown (magenta line).

The position estimate of the TN at the $j-$th iteration will be denoted as

$$\underline{\hat{u}}_j = (\hat{x}_j, \hat{y}_j, \bar{z}). \tag{10}$$

Similarly, from the definition of averaged range estimates $\hat{\rho}^K_{i,j}$ introduced in (5), it is also possible to define the projections of the averaged range estimates introduced in (5) on the plane $z = \bar{z}$ as

$$\hat{r}^K_{i,j} = \sqrt{\left(\hat{\rho}^K_{i,j}\right)^2 - h_i^2}. \tag{11}$$

Finally, the position estimates obtained by feeding the localization algorithm with averaged range estimates $\{\hat{r}^K_{i,j}\}^M_{i=1}$ are denoted as

$$\underline{\hat{v}}^K_j = (\hat{x}^K_j, \hat{y}^K_j, \bar{z}). \tag{12}$$

We remark that the j-th position estimate $\underline{\hat{v}}^K_j$ relies on the M averaged range estimates $\{\hat{\rho}^K_{i,j}\}^M_{i=1}$, and, therefore, on the $M \cdot K$ range estimates $\{\hat{\rho}_{i,h}\}^M_{i=1}$ with $h \in \{j, \ldots, j+K-1\}$. For this reason, the first position estimate $\underline{\hat{v}}^K_1$ can be obtained only after an initial phase during which K range estimates from each of the M APs are acquired. Once $\underline{\hat{v}}^K_1$ is evaluated, the position estimates that follow can be determined as soon as a new M-tuple of range estimates from the M APs is acquired. We remark that using the notation introduced for averaged range estimates it is also possible to consider the case without range averaging by setting $K = 1$ in (5).

IV. Two Relevant Localization Algorithms

Various range-based localization algorithms have been proposed in the literature (e.g., [26]) and they can all be integrated with the adopted JADE add-on module. This section describes the two algorithms used to obtain experimental results shown in next section, namely the TSML algorithm and the PSO-based algorithm. The starting point for the considered localization algorithms is (8), where the exact distances $\{r_i\}^M_{i=1}$ are replaced by their estimates $\hat{r}_{i,j}$. For the sake of simplicity, in the descriptions of localization algorithms we neglect the subscript j, which counts iterations, and we denote as $\{\hat{r}_i\}^M_{i=1}$ a generic set of range estimates at a given iteration. Using this notation, the non-linear system of equations (8) is replaced by

$$\begin{cases} (\hat{x} - x_1)^2 + (\hat{y} - y_1)^2 = \hat{r}_1^2 \\ \vdots \\ (\hat{x} - x_M)^2 + (\hat{y} - y_M)^2 = \hat{r}_M^2. \end{cases} \tag{13}$$

The system of equations (13) shows the equations of the M circumferences lying on the plane $z = \bar{z}$, centered in $\{(x_i, y_i)\}^M_{i=1}$ with radii $\{\hat{r}_{i,j}\}^M_{i=1}$. If the radii of such circumferences were the exact distances $\{r_i\}^M_{i=1}$, they would all intersect in the same point, which would correspond to the exact TN position. Instead, the radii of circumferences in (13) are range estimates, and they are affected by errors. Hence, circumferences do not intersect in a unique point and, therefore, proper localization algorithms are needed. In the following, we denote the solution of (13) as $\underline{\tilde{u}} = (\hat{x}, \hat{y})$.

A. Two-Stage Maximum-Likelihood Algorithm

In order to simplify the description of the algorithm, let us define the following quantity, related to the solution $\underline{\tilde{u}}$ of (13)

$$\hat{n} = \hat{x}^2 + \hat{y}^2. \tag{14}$$

Then, the system of equations (13) can be reformulated in matrix notation as

$$\underline{\underline{G}}_1 \, \underline{\hat{\omega}} = \underline{\hat{h}}_1, \tag{15}$$

where

$$\underline{\underline{G}}_1 = \begin{pmatrix} -2x_1 & -2y_1 & 1 \\ \vdots & \vdots & \vdots \\ -2x_M & -2y_M & 1 \end{pmatrix}$$
$$\underline{\hat{\omega}} = \begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{n} \end{pmatrix} \quad \underline{\hat{h}}_1 = \begin{pmatrix} \hat{r}^2_{1,j} - a_1^2 \\ \vdots \\ \hat{r}^2_{M,j} - a_M^2 \end{pmatrix} \tag{16}$$

and

$$a_i = \sqrt{x_i^2 + y_i^2} \tag{17}$$

We remark that (15) is not a linear system since the third element of vector $\underline{\hat{\omega}}$ depends on the first two elements according to (14). Neglecting this dependence, the solution $\underline{\hat{\omega}}$ of (15) is determined through a *Maximum-Likelihood* (*ML*) approach as

$$\underline{\hat{\omega}} = (\underline{\underline{G}}_1^{\intercal} \, \underline{\underline{W}}_1 \, \underline{\underline{G}}_1)^{-1} \underline{\underline{G}}_1^{\intercal} \, \underline{\underline{W}}_1 \, \underline{\hat{h}}_1, \tag{18}$$

where $\underline{\underline{W}}_1$ is a positive definite matrix [27]. For the sake of simplicity, in the implementation used to obtain discussed experimental results, matrix $\underline{\underline{W}}_1$ is equal to the identity matrix.

Once the solution $\underline{\hat{\omega}}$ of (15) is evaluated, the dependence of $\hat{n}$ on $\hat{x}$ and $\hat{y}$ can be taken into account by considering the following system of equations:

$$\underline{\underline{G}}_2 \, \underline{\hat{\phi}} = \underline{\hat{h}}_2, \tag{19}$$

where $\hat{\underline{\phi}} = (\hat{x}^2, \hat{y}^2)^{\intercal}$ and

$$\underline{\underline{G}}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \qquad \hat{\underline{h}}_2 = \begin{pmatrix} \hat{\omega}_1^2 \\ \hat{\omega}_2^2 \\ \hat{\omega}_3 \end{pmatrix}. \tag{20}$$

Let us remark that $\hat{\omega}_j$ in (20) denotes the $j-$th component of $\hat{\underline{\omega}}$. The solution of the rectangular system (19) can be determined, using a ML approach, as

$$\hat{\underline{\phi}} = (\underline{\underline{G}}_2^{\intercal}\, \underline{\underline{W}}_2\, \underline{\underline{G}}_2)^{-1} \underline{\underline{G}}_2^{\intercal}\, \underline{\underline{W}}_2\, \hat{\underline{h}}_2, \tag{21}$$

where $\underline{\underline{W}}_2$ is a positive definite matrix. In the implementation used to obtain discussed experimental results, matrix $\underline{\underline{W}}_2$ is set equal to the identity matrix, as done for $\underline{\underline{W}}_1$. Given the solution $\hat{\underline{\phi}}$, the estimated abscissa and ordinate at a generic iteration of the considered localization algorithm can be expressed as

$$\tilde{\underline{u}} = (\hat{x}, \hat{y}) = \underline{\underline{U}} \left(\sqrt{\hat{\phi}_1}, \sqrt{\hat{\phi}_2} \right)^{\intercal}, \tag{22}$$

where $\underline{\underline{U}} = \mathrm{diag}(\mathrm{sign}(\hat{\underline{\omega}}))$, and $\hat{\underline{\phi}} = (\hat{\phi}_1, \hat{\phi}_2)$. Finally, reintroducing the subscript j to denote the corresponding iteration, the estimated position of the TN at the $j-$th iteration can be written as

$$\hat{\underline{u}}_j = (\hat{x}_j, \hat{y}_j, \bar{z}). \tag{23}$$

The same localization algorithm can be applied also to the averaged range estimates $\hat{r}_{i,j}^K$ defined in (11). In this case, the initial system of equations can be obtained from (13) by substituting range estimates $\hat{r}_{i,j}$ with the averaged range estimates $\hat{r}_{i,j}^K$, namely,

$$\begin{cases} (\hat{x} - x_1)^2 + (\hat{y} - y_1)^2 = \left(\hat{r}_{1,j}^K\right)^2 \\ \vdots \\ (\hat{x} - x_M)^2 + (\hat{y} - y_M)^2 = \left(\hat{r}_{M,j}^K\right)^2. \end{cases} \tag{24}$$

The M equations in (24) represent the M circumferences lying on the plane $z = \bar{z}$, centered in $\{(x_i, y_i)\}_{i=1}^M$, with radii equal to the averaged range estimates $\{\hat{r}_{i,j}^K\}_{i=1}^M$. By applying the TSML algorithm to the system of equations in (24), it is possible to obtain position estimates evaluated using averaged distances and, in the following, such position estimates are denoted as

$$\hat{\underline{v}}_j^K = \left(\hat{x}_j^K, \hat{y}_j^K, \bar{z}\right). \tag{25}$$

B. The PSO-Based Algorithm

Observe that the system of equations (13) can be re-written in matrix notation as

$$\underline{1}\, \tilde{\underline{u}}^{\intercal} \tilde{\underline{u}} + \underline{\underline{A}}\, \tilde{\underline{u}} = \hat{\underline{k}}, \tag{26}$$

where $\underline{1}$ is the vector with M elements equal to 1, $\hat{\underline{k}}$ is a vector whose $i-$th element is $\hat{r}_i^2 - (x_i^2 + y_i^2)$, and $\underline{\underline{A}}$ is the following $M \times 2$ matrix

$$\underline{\underline{A}} = \begin{pmatrix} -2x_1 & -2y_1 \\ -2x_2 & -2y_2 \\ \vdots & \vdots \\ -2x_M & -2y_M \end{pmatrix}. \tag{27}$$

Also observe that the solution of the system of equations (26), and, hence, of the system of equations (13), can be reinterpreted as the solution of a related optimization problem. In detail, the solution $\tilde{u} = (\hat{x}, \hat{y})$ can be found as the solution of the following minimization problem:

$$\tilde{\underline{u}} = \arg\min_{\underline{u}}, F(\underline{u}) \tag{28}$$

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

$$F(\underline{u}) = ||\hat{k} - (\underline{1}\, \hat{\underline{u}}^{\intercal} \hat{\underline{u}} + \underline{\underline{A}}\, \hat{\underline{u}})||. \tag{29}$$

In order to solve the minimization problem (28), thus finding estimates for the abscissa and the ordinate of the TN, we proposed to use the PSO algorithm [28]. The PSO algorithm was first introduced in [29] and it considers the set of potential solutions of an optimization problem as a swarm of S particles which move through a *search space* according to proper rules. In detail, it is assumed that, at every instant, each particle is associated with a position in the search space and with a velocity. Positions and velocities of particles are iteratively updated according to proper rules, which are meant to move all particles towards the solution of the minimization problem, namely towards the position which minimizes the fitness function (29). The rules that are normally adopted to update positions and velocities are inspired by the rules which govern the behaviors of birds in swarms.

The use of PSO to support the localization of a TN works as follows. At initialization, the positions of the particles are randomly initialized in the search space, which, in our context, corresponds to the plane $z = \bar{z}$ where the abscissa and the ordinate of the TN are supposed to be situated. The initial positions are denoted as $\underline{x}^{(i)}(0)$, where $i \in \{1, \dots, S\}$ is the index of a generic particle and S is the number of particles. Similarly, the velocities of all the particles are randomly initialized and they are denoted as $\underline{w}^{(i)}(0)$, where $i \in \{1, \dots, S\}$ is the index of a generic particle. After this initialization phase, positions and velocities of all particles are updated at each iteration $t > 0$ to simulate interactions among individuals [30]. The position of the $i-$th particle at the $t-$th iteration is denoted as $\underline{x}^{(i)}(t)$ and its velocity is denoted as $\underline{w}^{(i)}(t)$. At each iteration, the velocity of the $i-$th particle is updated according to a specific rule expressed as the sum of three addends. In detail, the velocity of the $i-$th particle at the $(t+1)$-th iteration is [31]

$$\begin{aligned} \underline{w}^{(i)}(t+1) = \omega(t)\underline{w}^{(i)}(t) + c_1 R_1(t)(\underline{y}^{(i)}(t) - \underline{x}^{(i)}(t)) \\ + c_2 R_2(t)(\underline{y}(t) - \underline{x}^{(i)}(t)), \end{aligned} \tag{30}$$

where the following quantities [32] are used:

- $\underline{y}(t)$ is the best position globally reached so far;
- $\underline{y}^{(i)}(t)$ is the best position reached so far by the i-th particle;

- $\omega(t)$ is called *inertial factor*;
- c_1 is called *cognition* parameter;
- c_2 is called *social* parameter; and
- $R_1(t)$ and $R_2(t)$ are independent random variables uniformly distributed in $(0, 1)$.

The first addend in (30) is related to the velocity of the $i-$th particle at previous iteration t, which is weighed according to the inertial factor $\omega(t)$. In the second addend, the cognition parameter is a positive real parameter and the best position reached so far by the $i-$th particle can be expressed as

$$\underline{y}^{(i)}(t) = \arg\min_{\underline{z}\in X^{(i)}} F(\underline{z}), \tag{31}$$

where $X^{(i)} = \{\underline{x}^{(i)}(0), \ldots, \underline{x}^{(i)}(t)\}$. Hence, this second addend is meant to move each particle towards its best position reached so far. Finally, in the third addend, the social parameter is a positive real parameter and the best position reached so far by any particle in the swarm can be expressed as

$$\underline{y}(t) = \arg\min_{\underline{z}\in Y^{(i)}} F(\underline{z}), \tag{32}$$

where $Y^{(i)} = \{\underline{y}^{(1)}(t), \ldots, \underline{y}^{(S)}(t)\}$. The third addend is meant to move each particle towards the *global best position*, namely the position which corresponds to the lowest value of the fitness function among all those reached by any particle in the swarm [30]. The velocities computed with (30) are used to update the positions of particles at each iteration. Such updates are performed by adding the velocities evaluated in (30) to the previous positions of each particle, namely

$$\underline{x}^{(i)}(t+1) = \underline{x}^{(i)}(t) + \underline{w}^{(i)}(t+1). \tag{33}$$

The PSO algorithm is iterated until a termination condition is met. One of the possible termination conditions for the PSO algorithm is the reach of a maximum number of iterations. Once the execution of the algorithm is terminated, the solution corresponds to the position of the particle in the global best position, namely the position of the particle with the lowest value of the fitness function. This solution correspond to the estimated abscissa and ordinate of the TN.

The PSO algorithm outlined previously is used to solve the localization problem formulated in (28) and, hence, to solve the localization problem described in (13). The same algorithm can be applied also when the system of equations (24), where averaged range estimates $\hat{r}^K_{i,j}$ defined in (11) appear, is used. In this case, (24) can be re-written in matrix notation as

$$\underline{1}\,\underline{\tilde{u}}^{\mathsf{T}}\underline{\tilde{u}} + \underline{\underline{A}}\,\underline{\tilde{u}} = \underline{\hat{k}}_j \tag{34}$$

where $\underline{1}$ is the vector with M elements equal to 1, $\underline{\hat{k}}_j$ is a vector whose $i-$th element is $\left(\hat{r}^K_{i,j}\right)^2 - (x_i^2 + y_i^2)$, and $\underline{\underline{A}}$ is defined in (27). By applying the PSO algorithm to (34), it is possible to obtain position estimates evaluated using averaged distances and, in the following, such position estimates are denoted as

$$\underline{\hat{v}}^K_j = (\hat{x}^K_j, \hat{y}^K_j, \bar{z}). \tag{35}$$

Experimental results shown in the remaining of this paper are obtained with a population of $S = 40$ particles. The inertial factor is set to $\omega(t) = 0.5$ and the values of c_1 and c_2 are both set to 2, so that the average values of $c_1R_1(t)$ and of $c_2R_2(t)$ correspond to 1. The termination condition for the PSO algorithm corresponds to the reach of 50 iterations. These values proved to be effective for localization purposes [33]. Illustrative experimental results about the performance of the PSO-based algorithm are shown in next section.

V. Performance Evaluation

In the experimental campaign described in this section we consider three values for K, namely, $K = 1$ (i.e., no averaging); $K = 10$; and $K = 100$. The performance of discussed localization approaches is evaluated in terms of the distances between the true TN position and its estimates. In order to evaluate the performance of the discussed localization algorithms, let us define the distance error as

$$d_j = ||\underline{\hat{u}}_j - \underline{u}||. \tag{36}$$

Observe that, since we assume that the height of the TN is known, the third component of $\underline{\hat{u}}_j$ is equal to the third component of the vector $\underline{u}$ which represents the true TN position. Therefore, (36) represents the projection of the distance error on the plane $z = \bar{z}$. The definition of the distance error (36) allows introducing the maximum value of the distance errors, which can be denoted as

$$d_{\max} = \max_{j\in\{1,\ldots,L\}} d_j. \tag{37}$$

Let us also introduce the average value of the distance error, which can be expressed as

$$d_{\text{avg}} = \frac{1}{L}\sum_{j=1}^{L} d_j. \tag{38}$$

Finally, the standard deviation of the distance error is

$$\sigma_d = \sqrt{\frac{1}{L}\sum_{j=1}^{L}(d_j - d_{\text{avg}})^2}. \tag{39}$$

Analogous values relative to position estimates $\underline{\hat{v}}^K_j$ obtained using averaged range estimates can be defined. In detail, let us define

$$\delta^K_j = ||\underline{\hat{v}}^K_j - \underline{u}|| \tag{40}$$

which represents the distance error on the plane $z = \bar{z}$ between the true TN position and its estimate in the $j-$th iteration obtained using averaged range estimates over K consecutive range acquisitions. The definition of δ^K_j allows computing the maximum value of the distance error as

$$\delta^K_{\max} = \max_{j\in\{1,\ldots,L\}} \delta^K_j, \tag{41}$$

and the average value of the distance error as

$$\delta^K_{\text{avg}} = \frac{1}{L}\sum_{j=1}^{L} \delta^K_j. \tag{42}$$

Finally, the standard deviation of the distance error is

$$\sigma_{\delta}^{K} = \sqrt{\frac{1}{L}\sum_{j=1}^{L}(\delta_j^K - \delta_{\text{avg}}^K)^2}. \tag{43}$$

Observe that the values in (37), (38), and (39) can be equally defined using the more general notation in (41), (42), and (43), respectively, for $K = 1$.

In order to assess the accuracy of the localization algorithms previously described, we performed an experimental campaign in an illustrative indoor scenario which consists of a square room whose sides are 4 m long. The considered scenario is shown in Fig. 2, where $M = 3$ APs are shown (blue squares). Observe that $M = 3$ is the minimum number of APs which allows the application of described localization algorithms. The coordinates of the APs are denoted as $\{\text{AP}_i\}_{i=1}^{3}$ and they are positioned in the room in such a way that, in a proper coordinate system, they can be expressed, in meters, as

$$\begin{aligned} \text{AP}_1 &= (0,0,3)^{\mathsf{T}} \\ \text{AP}_2 &= (0,4,3)^{\mathsf{T}} \\ \text{AP}_3 &= (4,4,3)^{\mathsf{T}}. \end{aligned} \tag{44}$$

In Fig. 2, three different TN positions are also shown (red stars) and their position in the same coordinate system can be expressed in meters as

$$\begin{aligned} \underline{u}_1 &= (1,1,1)^{\mathsf{T}} \\ \underline{u}_2 &= (1,2,1)^{\mathsf{T}} \\ \underline{u}_3 &= (2,2,1)^{\mathsf{T}}. \end{aligned} \tag{45}$$

Using the described configuration of fixed APs, the three different TN positions are estimated. Results of such position estimates are discussed in the remaining of this section, using both localization algorithms introduced in previous section, on the basis of the distance error discussed above. Let us remark that, even if in the considered scenario all APs are placed at the same height (i.e., 3 m), and all TNs are placed at the same height (i.e., 1 m), both proposed localization algorithms are general and they do not require that APs share the same height. Moreover, different heights for the TNs could also be considered, provided that they are known, so that the values of h_i defined in (6) can be computed.

In the remaining of this section, relevant comments on the localization of the three TNs are presented. In all scenarios, the number of iterations in set equal to $L = 100$. Hence, the average value and the standard deviations of distance errors is based on 100 position estimates. In the following figures and tables, in order to distinguish between position estimates obtained using the two algorithms, we add superscript (T) and (P) to denote position estimates derived using the TSML and the PSO-based algorithm, respectively.

A. First Scenario

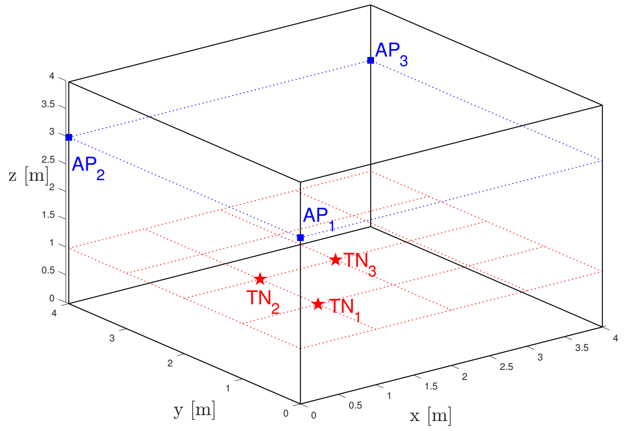

Fig. 2. The positions of the three considered APs (blue squares) and three different TN positions (red stars) are shown.

We start by considering the TN denoted as TN_1 in Fig. 2, whose coordinates are $\underline{u}_1 = (1,1,1)^T$. Since the coordinates of the APs are also known, the true distances $\{\rho_i\}_{i=1}^{3}$ between the i-th AP and the TN can be computed. In detail, from (44) and (45), the values of $\{\rho_i\}_{i=1}^{3}$ can be computed as

$$\rho_1 \simeq 2.45 \text{ m} \qquad \rho_2 \simeq 3.74 \text{ m} \qquad \rho_3 \simeq 4.69 \text{ m}.$$

The projections $\{r_i\}_{i=1}^{3}$ of the range estimates $\{\rho_i\}_{i=1}^{3}$ on the plane $z = 1$ m can be also computed according to the Pythagorean theorem. Simple algebraic manipulations show that the values of $\{r_i\}_{i=1}^{3}$ are

$$r_1 \simeq 1.41 \text{ m} \qquad r_2 \simeq 3.16 \text{ m} \qquad r_3 \simeq 4.24 \text{ m}.$$

In order to apply the discussed algorithms, we acquire range estimates from each AP to have: 100 position estimates obtained without range averaging; 100 position estimates obtained by averaging over $K = 10$ consecutive range estimates; and 100 position estimates obtained by averaging over $K = 100$ consecutive range estimates.

Fig. 3 shows the true position of TN_1 on the plane $z = 1$ (red star). In the same figure, the projections of the position estimates on the plane $z = 1$ (black circles) obtained without averaging are also shown, together with the projections of the position estimates (magenta crosses) obtained by averaging over 10 consecutive range estimates from each AP, and the projections of the position estimates (green triangles) obtained by averaging over 100 consecutive range estimates from each AP. From Fig. 3 it can be observed that, as expected, the position estimates are closer to each other as K increases. This is in agreement with the intuitive idea that averaging range estimates over large values of K leads, on average, to more accurate values of $\hat{r}_{i,j}^{K}$. The more accurate are the range estimates used to feed the localization algorithm, the more accurate are the obtained position estimates. Fig. 4 shows the *Cumulative Distribution Functions* (*CDFs*) of the distance errors without range averaging (black line), with $K = 10$ (dashed magenta line), and with $K = 100$ (dash-dotted green line). As intuitively expected from Fig. 3, the larger is K, the steepest is the graph of the CDF.

Fig. 5 shows the true position of TN_1 on the plane $z = 1$ (red star), together with: the projections of position estimates

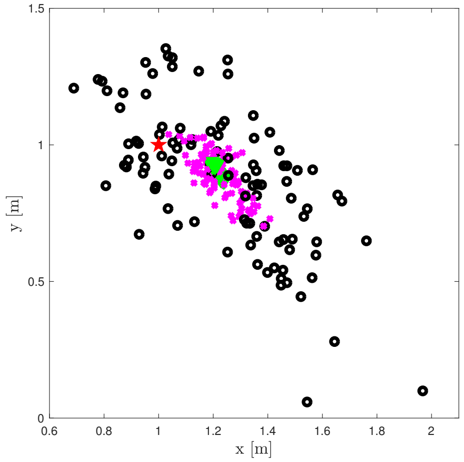

Fig. 3. Projections of the position estimates of TN_1 in Fig. 2 obtained using the TSML algorithm on the plane $z = 1$: *(i).* without range averaging (black circles); *(ii)* with $K = 10$ (magenta crosses); and *(iii)* with $K = 100$ (green triangles). The true TN position (red star) is also shown.

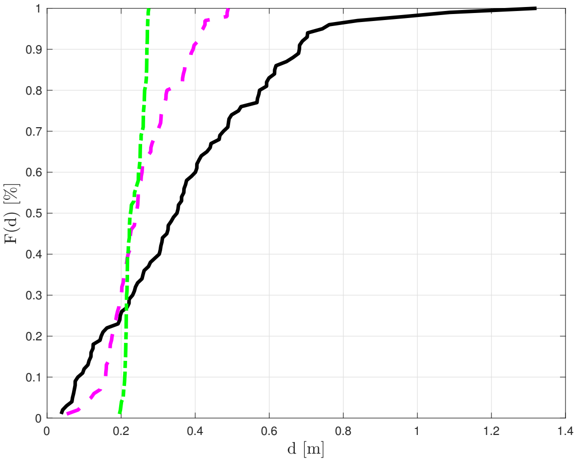

Fig. 4. The cumulative distribution function of distance errors from Fig. 3: *(i)* without range averaging (black line); *(ii)* with $K = 10$ (dashed magenta line), and *(iii)* with $K = 100$ (dash-dotted green line), relative to the position estimates of TN_1 obtained using the TSML algorithm.

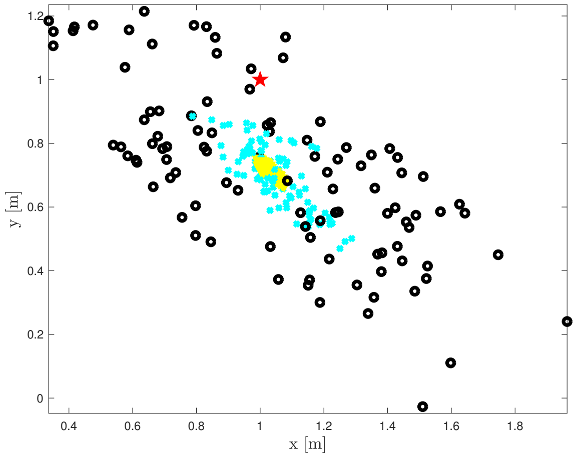

Fig. 5. Projections of the position estimates of TN_1 in Fig. 2 obtained using the PSO algorithm on the plane $z = 1$: *(i).* without range averaging (black circles); *(ii)* with $K = 10$ (cyan crosses); and *(iii)* with $K = 100$ (yellow triangles). The true TN position (red star) is also shown.

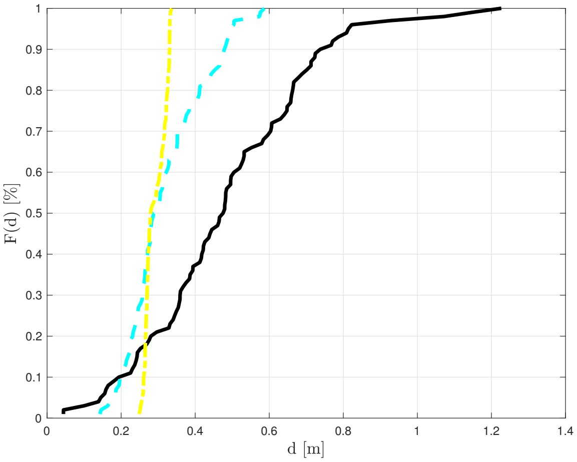

Fig. 6. The cumulative distribution function of distance errors from Fig. 5: *(i)* without range averaging (black line); *(ii)* with $K = 10$ (dashed cyan line), and *(iii)* with $K = 100$ (dash-dotted yellow line), relative to the position estimates of TN_1 obtained using the PSO algorithm.

on the plane $z = 1$ (black circles) obtained without range averaging; the projections of position estimates (cyan crosses) obtained by averaging over 10 consecutive range estimates from each AP; and the projections of position estimates (yellow triangles) obtained by averaging 100 consecutive range estimates from each AP. As observed when using the TSML algorithm, the position estimates are closer to the TN position as K increases. Hence, also when considering the PSO-based algorithm, more accurate range estimates lead to more accurate position estimates. Fig. 6 shows the CDFs of distance errors without range averaging (black line), with $K = 10$ (dashed cyan line), and with $K = 100$ (dash-dotted yellow line). As expected, larger values of K correspond to steeper CDFs.

Table I shows the values of the maximum distances and of the average distances between the considered TN and its estimates, and the values of standard deviations of distance errors for $K = 1$, $K = 10$, and $K = 100$. It can be observed that when localization is performed using the TSML algorithm,

TABLE I. Values of the Maximum Distance Error (First Row), of the Average Distance Error (Second Row), and of the Standard Deviation of the Distance Error (Third Row) are Shown, for Different Values of K, Relative to Position Estimates of TN_1 Obtained with the TSML Algorithm and with the PSO-based Algorithm, Respectively

TN_1	TSML			PSO		
K	1	10	100	1	10	100
δ_{max}^{K} [m]	1.32	0.49	0.27	1.23	0.59	0.34
δ_{avg}^{K} [m]	0.38	0.26	0.24	0.49	0.32	0.29
σ_{δ}^{K} [m]	0.24	0.09	0.02	0.23	0.10	0.03

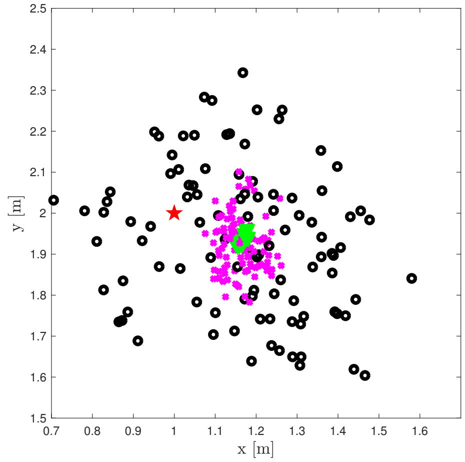

Fig. 7. Projections of the position estimates of TN_2 in Fig. 2 obtained using the TSML algorithm on the plane $z = 1$: *(i)* without range averaging (black circles); *(ii)* with $K = 10$ (magenta crosses); and *(iii)* with $K = 100$ (green triangles). The true TN position (red star) is also shown.

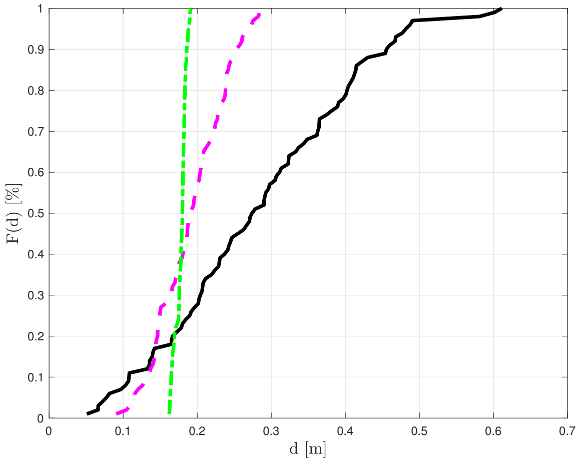

Fig. 8. The cumulative distribution function of distance errors from Fig. 7: *(i)* without range averaging (black line); *(ii)* with $K = 10$ (dashed magenta line), and *(iii)* with $K = 100$ (dash-dotted green line), relative to the position estimates of TN_2 obtained using the TSML algorithm.

the values of the maximum distance between the considered TN and its estimates decrease as K increases. In detail, without range averaging the values of the maximum distance between the considered TN and its estimates is equal to 1.32 m and it decreases to 0.49 m when $K = 10$ and to 0.27 m when $K = 100$. Analogous considerations hold when analyzing results relative to the PSO-based algorithm. In this case, the value of the maximum distance between the considered TN and its estimates without range averaging equals 1.23 m, and it is slightly smaller than that evaluated when considering the TSML algorithm. The values of the maximum distance between the considered TN and its estimates are equal to 0.59 m and to 0.34 m when $K = 10$ and $K = 100$, respectively, and they are slightly higher than those obtained by applying TSML algorithm. Form the fourth row of Table I it can be observed that also the values of the average distance between the considered TN and its estimates decrease as K increases, starting from 0.38 m when no range averaging is considered to 0.24 m when $K = 100$ is considered. The values of the average distance δ_{avg}^{K} between the considered TN and its estimates when using the PSO-based algorithm are comparable to those obtained when using the TSML algorithm, and they are equal to 0.49 m, when range averages are not performed, and to 0.32 m and 0.29 m when $K = 10$ and $K = 100$, respectively. Finally, Table I shows that the values of the standard deviations of distance errors σ_{δ}^{K} also decrease when K increases. This results was expected also from Fig. 4 and Fig. 6, from which it is evident that increasing K not only reduces the distances between the position estimates and the TN, but it also reduces the distances between different position estimates because it alleviates the influence of acquisition errors. For the same choice of K, the values of the standard deviation evaluated using the TSML algorithm and using the PSO-based algorithm are similar and their order of magnitude is 0.2 m without range averaging and 0.02 m when $K = 100$.

B. Second Scenario

We now consider the TN positioned in the point denoted as TN_2 in Fig. 2, whose coordinates are denoted as $\underline{u}_2$ in (45). In this case, the true distances $\{\rho_i\}_{i=1}^{3}$ between the i-th AP and the TN are

$$\rho_1 \simeq 3 \text{ m} \qquad \rho_2 \simeq 3 \text{ m} \qquad \rho_3 \simeq 4.12 \text{ m}.$$

The projections of the range estimates on the plane $z = 1$ m can be evaluated, according to the Pythagorean theorem, as

$$r_1 \simeq 2.23 \text{ m} \qquad r_2 \simeq 2.23 \text{ m} \qquad r_3 \simeq 3.60 \text{ m}.$$

In order to estimate the position of TN_2, range estimates from the APs are taken in order to have: 100 position estimates obtained without range averaging; 100 position estimates obtained by averaging over $K = 10$ consecutive range estimates; and 100 position estimates obtained by averaging over $K = 100$ consecutive range estimates.

In Fig. 7, the true position of TN_2 on the plane $z = 1$ (red star) is shown. Fig. 7 also shows the projections of the position estimates on the plane $z = 1$ (black circles) obtained without range averaging. Moreover, the projections of the position estimates (magenta crosses) obtained by averaging over 10 consecutive range estimates from each AP, and the projections of the position estimates (green triangles) obtained by averaging over 100 consecutive range estimates from each AP are shown. Fig. 7 shows that the distance among the TN position and its estimates decreases as K increases. This is motivated by the fact that large values of K lead to more precise averaged range estimates, which allow more accurate position estimates to be derived.

Fig. 8 shows the CDFs of the distance errors without range averaging (black line), with $K = 10$ (dashed magenta line), and with $K = 100$ (dash-dotted green line). As when considering TN_1, the steepness of the CDF increases as K also increases.

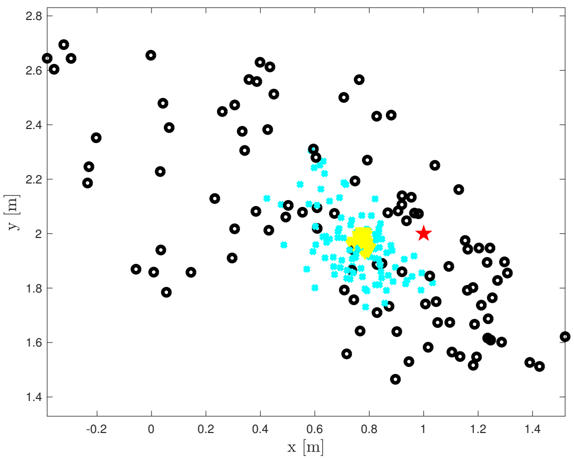

Fig. 9. Projections of the position estimates of TN$_2$ in Fig. 2 obtained using the PSO algorithm on the plane $z = 1$: *(ii)* without range averaging (black circles); *(ii)* with $K = 10$ (cyan crosses); and *(iii)* with $K = 100$ (yellow triangles). The true TN position (red star) is also shown.

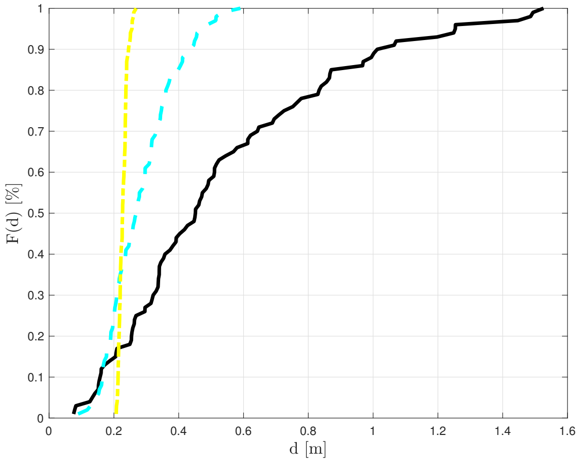

Fig. 10. The cumulative distribution function of distance errors from Fig. 9: *(i)* without range averaging (black line); *(ii)* with $K = 10$ (dashed cyan line), and *(iii)* with $K = 100$ (dash-dotted yellow line), relative to the position estimates of TN$_2$ obtained using the PSO algorithm.

TABLE II. Values of the Maximum Distance Error (First Row), of the Average Distance Error (Second Row), and of the Standard Deviation of the Distance Error (Third Row) are Shown, for Different Values of K, Relative to Position Estimates of TN$_2$ Obtained with the TSML Algorithm and with the PSO-based Algorithm, Respectively

TN$_2$	TSML			PSO		
K	1	10	100	1	10	100
$\delta^K_{\max}$ [m]	0.61	0.29	0.19	1.53	0.59	0.27
δ^K_{avg} [m]	0.28	0.19	0.18	0.54	0.28	0.23
σ^K_δ [m]	0.13	0.05	0.008	0.35	0.10	0.01

Fig. 9 shows the true position of TN$_2$ on the plane $z = 1$ (red star), together with, the projections of the position estimates (black circles) obtained without range averaging, the projections of the position estimates (cyan crosses) obtained with $K = 10$, and the projections of the position estimates (yellow triangles) obtained with $K = 100$. As observed when using the TSML algorithm, the accuracy of the position estimates increases as K increases. Hence, also when considering the PSO-based algorithm, more accurate range estimates lead to more accurate position estimates.

Fig. 10 shows the CDFs of the distance errors without range averaging (black line), with $K = 10$ (dashed cyan line), with $K = 100$ (dash-dotted yellow line). As expected from Fig. 9, the larger is K, the steepest is the graph of the CDF.

Table II shows the values of the maximum distances and of the average distances between the considered TN and its estimates, and the values of the standard deviations of the distance errors for $K = 1$, $K = 10$, and $K = 100$. As when considering the first scenario, all such values decrease as the number of range averages K increases. Table II also shows that with no range estimate averaging, the maximum distance and the average distance between the considered TN and its estimates when using the PSO-based algorithm correspond to the double of the analogous values obtained with the TSML algorithm. When $K = 10$ and $K = 100$, values obtained with the TSML algorithm are more similar to each other, even though the values obtained with the latter are slightly higher than those obtained with the former. Concerning the standard deviations of the distance error, it can be observed that, as in the first scenario, the values corresponding to $K = 100$ are one order of magnitude lower than those obtained when $K = 1$.

C. Third Scenario

Finally, let us now consider the TN positioned in the middle of the room, denoted as TN$_3$ in Fig. 2, whose coordinates are denoted as $\underline{u}_3$ in (45). In this case, the true distances $\{\rho_i\}_{i=1}^3$ between the i-th AP and the TN, expressed in meters, are

$$\rho_1 = \rho_2 = \rho_3 \simeq 3.46 \text{ m}.$$

Since the values of $\{h_i\}_{i=1}^3$ are 2 m, the projections of the distances on the plane $z = 1$ m can be computed as

$$r_1 = r_2 = r_3 \simeq 2.83 \text{ m}.$$

As in previous scenarios, range estimates from each of the three APs are acquired in order to have: 100 position estimates obtained without range averaging; 100 position estimates obtained by averaging over $K = 10$ consecutive range estimates; and 100 position estimates obtained by averaging over $K = 100$ consecutive range estimates.

Fig. 11 shows the projection of the true position of TN$_3$ on the plane $z = 1$ (red star). In the same figure, the projections of the position estimates (black circles) obtained without range averaging, the projections of the position estimates (magenta crosses) obtained by averaging over 10 consecutive range estimates from each AP, and the projections of the position estimates (cyan triangles) obtained by averaging over 100 consecutive range estimates from each AP, are also shown. As in the previous cases, Fig. 11 shows that the distance between position estimates and the TN decreases as K increases.

TABLE III. Values of the Maximum Distance Error (First Row), of the Average Distance Error (Second Row), and of the Standard Deviation of the Distance Error (Third Row) are Shown, for Different Values of K, Relative to Position Estimates of TN_3 Obtained with the TSML Algorithm and with the PSO-based Algorithm, Respectively

TN_3	TSML			PSO		
K	1	10	100	1	10	100
δ^K_{max} [m]	0.64	0.31	0.16	0.88	0.47	0.33
δ^K_{avg} [m]	0.31	0.18	0.15	0.43	0.34	0.32
σ^K_{δ} [m]	0.15	0.05	0.006	0.23	0.05	0.004

Fig. 12 shows the CDFs of the distance errors without range averaging (black line), with $K = 10$ (dashed magenta line), and with $K = 100$ (dash-dotted green line), which is steeper as K increases.

Fig. 13 shows the true position of TN_3 on the plane $z = 1$ (red star), together with the projections of the position estimates on the plane $z = 1$ (black circles) obtained without range averaging, the projections of the position estimates (cyan crosses) obtained by averaging over 10 consecutive range estimates from each AP, and the projections of the position estimates (yellow triangles) obtained by averaging over 100 consecutive range estimates from each AP. As in previous scenarios, the position estimates are closer to each other and to the true TN position as K increases. Hence, also when considering the PSO-based algorithm, more accurate range estimates lead to more accurate position estimates.

Fig. 14 shows the CDFs of the distance errors without range averaging (black line), with $K = 10$ (dashed cyan line), and with $K = 100$ (dash-dotted yellow line). Once again, the larger is K, the steepest is the graph of the CDF.

Table III shows the values of the maximum distances and of the average distances between the considered TN and it estimates, and the values of the standard deviations of the the distance errors for $K = 1$, $K = 10$, and $K = 100$.

As when considering previous scenarios, all such values decrease as the number of range averages K increases. Moreover, for the same choice of K values obtained with the TSML algorithm are of the same order of magnitude than those obtained with the PSO-based algorithm, even though the values obtained with the latter are slightly higher than those obtained with the former. Concerning the standard deviations of the distance error, it can be observed that, as in previous scenarios, the values corresponding to $K = 100$ are one order of magnitude lower than those obtained when $K = 1$. It is worth observing that, according to results shown in Tables I, II and III, the performance of both algorithms are similar in all scenarios.

VI. Conclusion

This paper presented an experimental evaluation of two approaches to indoor localization which both use ordinary WiFi signaling with no dedicated localization infrastructure. In both discussed approaches, agents acquire range estimates from the APs of the WiFi infrastructure, and they use such estimates to obtain real-time information on the position of the smart appliances which hosts them.

The results obtained in the presented experimental campaign are meant to give a quantitative assessment on the performance of WiFi-based indoor localization, and they show that the level of accuracy of WiFi-based localization can be increased by a proper pre-processing of acquired range estimates. Obtained results show that agents can reach a localization accuracy of less than 1 m, thus making the proposed approach adequate for many application scenarios. In particular, such an accuracy is sufficient to support social games in large environments like shopping malls, waiting areas of airports and train stations, and covered markets in historical areas of towns. It is worth noting that presented results are valid under the assumption that the smart appliance does not move, or that it moves slowly with respect to range acquisition rate. Such an assumption does not necessarily hold for social games, and further investigation on dynamic scenarios is in progress.

References

[1] Z. Farid, R. Nordin, and M. Ismail, "Recent advances in wireless indoor localization techniques and system," *Journal of Computer Networks and Communications*, vol. 2013, 2013.

[2] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low cost outdoor localization for very small devices," *IEEE Personal Communications*, vol. 7, no. 5, pp. 28–34, October 2000.

[3] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1067–1080, November 2007.

[4] S. Monica and F. Bergenti, "A comparison of accurate indoor localization of static targets via WiFi and UWB ranging," in *Trends in Practical Applications of Scalable Multi-Agent Systems, the PAAMS Collection, 14^{th} International Conference on Practical Applications of Agents and Multi-Agent Systems* (PAAMS 2016)*, Special Session on Agents and Mobile Devices* (AM), Sevilla, Spain, June 2016, pp. 111–123.

[5] S. Gezici and H. V. Poor, "Position estimation via Ultra-Wide-Band signals," *Proceedings of the IEEE*, vol. 97, no. 2, pp. 386–403, February 2009.

[6] F. Bellifemine, F. Bergenti, G. Caire, and A. Poggi, "JADE – A Java Agent DEvelopment framework," in *Multi-Agent Programming*. Springer, 2005, pp. 125–147.

[7] G. Adorni, F. Bergenti, A. Poggi, and G. Rimassa, "Enabling FIPA agents on small devices," in 5^{th} *International Workshop on Cooperative Information Agents (CIA 2001)*, 2001, pp. 248–257.

[8] F. Bergenti, G. Caire, and D. Gotta, "Agents on the move: JADE for Android devices," in *Proceedings of the* 15^{th} *Workshop From Objects to Agents*, ser. CEUR Workshop Proceedings, vol. 1260. RWTH Aachen, 2014.

[9] F. Bergenti, "An introduction to the JADEL programming language," in *Procs. of the IEEE 26^{th} Int'l Conference on Tools with Artificial Intelligence (ICTAI)*, 2014, pp. 974–978.

[10] F. Bergenti, E. Iotti, S. Monica, and A. Poggi, "Interaction protocols in the JADEL programming language," in *International Workshop on Programming based on Actors, Agents, and Decentralized Control* (AGERE) *at the ACM SIGPLAN conference on Systems, Programming, Languages, and Applications: Software for Humanity* (SPLASH), 2016.

[11] F. Bergenti, E. Iotti, S. Monica, and A. Poggi, "Agent-oriented model-driven development for JADE with the JADEL programming language," *Computer Languages, Systems and Structures*, vol. 50, pp. 142–158, 2017.

[12] A. Poggi and F. Bergenti, "Developing smart emergency applications with multi-agent systems," *Int. J. E-Health Med. Commun.*, vol. 1, no. 4, pp. 1–13, 2010.

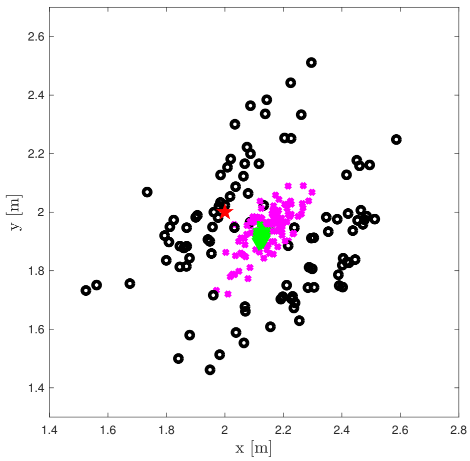

Fig. 11. Projections of the position estimates of TN_3 in Fig. 2 obtained using the TSML algorithm on the plane $z = 1$: *(ii)* without range averaging (black circles); *(ii)* with $K = 10$ (magenta crosses); and *(iii)* with $K = 100$ (green triangles). The true TN position (red star) is also shown.

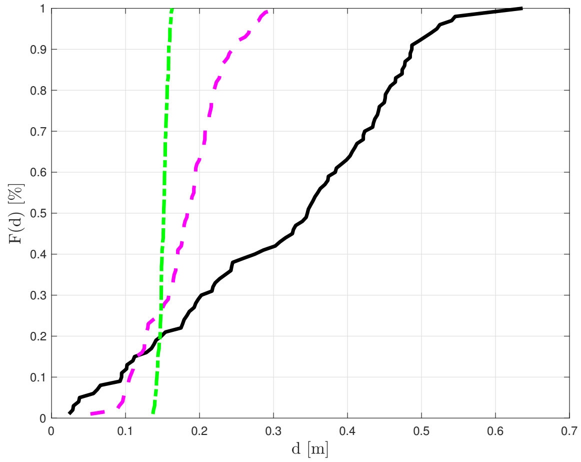

Fig. 12. The cumulative distribution function of distance errors from Fig. 11: *(i)* without range averaging (black line); *(ii)* with $K = 10$ (dashed magenta line), and *(iii)* with $K = 100$ (dash-dotted green line), relative to the position estimates of TN_3 obtained using the TSML algorithm.

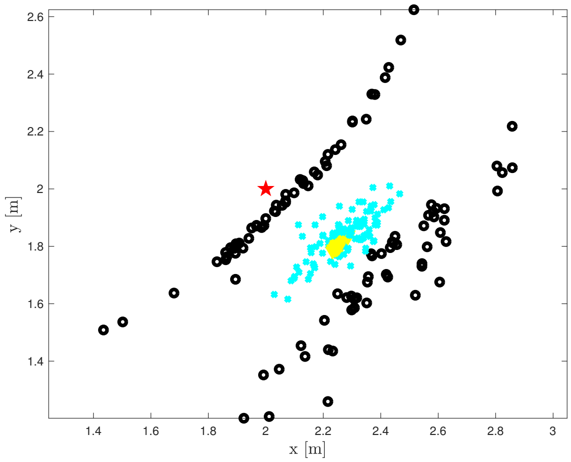

Fig. 13. Projections of the position estimates of TN_3 in Fig. 2 obtained using the PSO algorithm on the plane $z = 1$: *(ii)* without range averaging (black circles); *(ii)* with $K = 10$ (cyan crosses); and *(iii)* with $K = 100$ (yellow triangles). The true TN position (red star) is also shown.

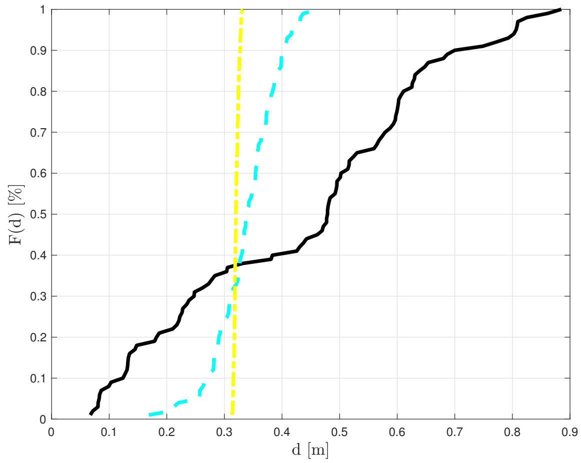

Fig. 14. The cumulative distribution function of distance errors from Fig. 13: *(i)* without range averaging (black line); *(ii)* with $K = 10$ (dashed cyan line), and *(iii)* with $K = 100$ (dash-dotted yellow line), relative to the position estimates of TN_3 obtained using the PSO algorithm.

[13] F. Bergenti, E. Franchi, and A. Poggi, "Agent-based social networks for enterprise collaboration," in 20th *IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 2011, pp. 25–28.

[14] F. Bergenti, G. Caire, and D. Gotta, "Large-scale network and service management with WANTS," in *Industrial Agents: Emerging Applications of Software Agents in Industry*. Elsevier, 2015, pp. 231–246.

[15] F. Bergenti, G. Caire, and D. Gotta, "An overview of the AMUSE social gaming platform," in *Proceedings of the* 14th *Workshop From Objects to Agents*, ser. CEUR Workshop Proceedings, vol. 1099. RWTH Aachen, 2013.

[16] F. Bergenti and S. Monica, "Location-Aware Social Gaming with AMUSE," in *Trends in Practical Applications of Scalable Multi-Agent Systems, the PAAMS Collection,* 14th *International Conference on Practical Applications of Agents and Multi-Agent Systems* (PAAMS 2016), ser. LNCS, vol. 9662, 2016, pp. 36–47.

[17] S. Monica and F. Bergenti, "Location-aware JADE agents in indoor scenarios," in *Proceedings of the* 16th *Workshop From Objects to Agents*, ser. CEUR Workshop Proceedings, vol. 1382. RWTH Aachen, 2015, pp. 103–108.

[18] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, "Range-free localization schemes for large scale sensor networks," in *Proceedings of the* 9th *Annual International Conference on Mobile Computing and Networking* (MobiCom), San Diego, CA, September 2003, pp. 81–95.

[19] D. Dardari and R. D'Errico, "Passive ultrawide bandwidth RFID," in *Proceedings of the IEEE Global Telecommunications Conference* (GLOBECOM '08), New Orleans, LA, December 2008, pp. 1–6.

[20] Z. Sahinoglu, S. Gezici, and I. Guvenc, *Ultra-wideband positioning systems: Theoretical limits, ranging algorithms and protocols*. Cambridge, U.K.: Cambridge University Press, 2008.

[21] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the nodes," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54–69, July 2005.

[22] Y. Chan and K. C. Ho, "A simple and efficient estimator for hyperbolic location," *IEEE Transactions on Signal Processing*, vol. 42, no. 8, pp.

1905–1915, August 1994.

[23] S. Monica and G. Ferrari, “A swarm-based approach to real-time 3D indoor localization: Experimental performance analysis,” *Applied Soft Computing*, vol. 43, pp. 489–497, June 2016.

[24] S. Monica and F. Bergenti, “An experimental evaluation of agent-based indoor localization,” in *The Science and Information Computing Conference, technically sponsored by IEEE.* IEEE Press, July 2017, pp. 638–646.

[25] S. Monica and F. Bergenti, “Experimental evaluation of agent-based localization of smart appliances,” in *Proceedings of the European Conference on Multi-Agent Systems (EUMAS 2016)*, ser. LNAI, vol. 10207, 2017.

[26] S. Monica and G. Ferrari, “Accurate indoor localization with UWB wireless sensor networks,” in *Proceedings of the* 23^{rd} *IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises* (WETICE 2014). IEEE Press, 2014, pp. 287–289.

[27] K. C. Ho, X. Lu, and L. Kovavisaruch, “Source localization using TDOA and FDOA measurements in the presence of receiver location errors: Analysis and solution,” *IEEE Transactions on Signal Processing*, vol. 55, no. 2, pp. 684–696, February 2007.

[28] S. Monica and G. Ferrari, “Swarm intelligent approaches to auto-localization of nodes in static UWB networks,” *Applied Soft Computing*, vol. 25, pp. 426–434, December 2014.

[29] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of the IEEE International Conference on Neural Networks* (ICNN), Perth, Australia, November 1995, pp. 1942–1948.

[30] R. Poli, J. Kennedy, and T. Blackwell, “Particle swarm optimization,” *Swarm Intelligence Journal*, vol. 1, no. 1, pp. 33–57, June 2007.

[31] Y. Shi and R. Eberhart, “A modified particle swarm optimizer,” in *Proceedings of the IEEE International Conference on Evolutionary Computation* (ICEC), Washington, DC, July 1999, pp. 69–73.

[32] R. Eberhart and J. Kennedy, “A new optimizer using particles swarm theory,” in *Proceedings of the* 6^{th} *International Symposium on Micro Machine and Human Science* (MHS), Nagoya, Japan, October 1995, pp. 39–43.

[33] S. Monica and G. Ferrari, “Particle swarm optimization for auto-localization of nodes in wireless sensor networks,” in *Proceedings of the* 11^{th} *International Conference on Adaptive and Natural Computing Algorithms* (ICANNGA ’13), ser. LNCS, vol. 7824, 2013, pp. 456–465.