



## Article

# An Algorithm for Accurate and Robust Indoor Localization Based on Nonlinear Programming

Stefania Monica <sup>\*,†</sup>  and Federico Bergenti <sup>†</sup> 

Dipartimento di Scienze Matematiche, Fisiche e Informatiche, Università degli Studi di Parma,  
43124 Parma, Italy; federico.bergenti@unipr.it

\* Correspondence: stefania.monica@unipr.it; Tel.: +39-0521-906-900

† These authors contributed equally to this work.

Received: 27 October 2019; Accepted: 18 December 2019; Published: 1 January 2020



**Abstract:** The study of techniques to estimate the position of mobile devices with a high level of accuracy and robustness is essential to provide advanced location based services in indoor environments. An algorithm to enable mobile devices to estimate their positions in known indoor environments is proposed in this paper under the assumption that fixed anchor nodes are available at known locations. The proposed algorithm is specifically designed to be executed on the mobile device whose position is under investigation, and it allows the device to estimate its position within the environment by actively measuring distance estimates from the anchor nodes. In order to reduce the impact of the errors caused by the arrangement of the anchor nodes in the environment, the proposed algorithm first transforms the localization problem into an optimization problem, and then, it solves the derived optimization problem using techniques inspired by nonlinear programming. Experimental results obtained using ultra-wide band signaling are presented to assess the performance of the algorithm and to compare it with reference alternatives. The presented experimental results confirm that the proposed algorithm provides an increased level of accuracy and robustness with respect to two reference alternatives, regardless of the position of the anchor nodes.

**Keywords:** localization as optimization; indoor localization; localization algorithms

## 1. Introduction and Motivation

The possibility of computing accurate position estimates of mobile devices in indoor environments is essential to increase the level of personalization of services offered to users (e.g., [1]). Among the plethora of applications that can take advantage of accurate and robust indoor localization, it is worth mentioning educational games designed to increase the interactivity of visits to exhibitions and museums (e.g., [2,3]), applications to support the automation of work in industrial warehouses (e.g., [4]), and advanced services related to location based social networks (e.g., [5]), which can be offered in large indoor areas like shopping malls, train stations, and airports. Currently, the literature documents solutions to specific indoor problems, such as presence analysis using crowdsensing [6] and identification of stop-by behaviors [7]. However, the possibility of enabling a mobile device to estimate its position accurately in an indoor environment is still an open problem (e.g., [8,9]), even if specialized technologies, like Ultra-Wide Band (UWB) (e.g., [10]), are now easily implemented.

In this paper, the attention is focused on the problem of enabling a mobile device, denoted as the Target Node (TN), to compute accurately and robustly its position in a known indoor environment in which fixed network nodes, denoted as Anchor Nodes (ANs), are installed at known locations. In the considered scenarios, mobile devices can actively measure the distances from visible ANs using one of the available ranging technologies, like UWB, which is the technology that was actually used to obtain the experimental results presented in Section 4. Note that the proposed method to compute the position

of a TN is not restricted to work with UWB, and preliminary experiments that use WiFi signaling were also performed (e.g., [11,12]). The accepted tolerance of the obtained position estimates depends on the considered application, but an average error of 1 m is normally considered acceptable for many application scenarios, while experimental results discussed in Section 4 show that such an accuracy is feasible using UWB. With minor loss of generality, the considered indoor environment is assumed to be composed of possibly overlapping rectangular cuboids, called boxes. Such a choice allows focusing on single boxes when searching for the TN in the considered environment. Observe that the minimum bounding box containing the considered environment can be used if the environment cannot be split into boxes, and in such cases, the computed estimates of the position of the TN should be filtered accordingly.

One of the problems that makes accurate indoor localization difficult in the considered scenarios is that the positions of the ANs in the environment strongly impact the accuracy of ordinary localization algorithms (e.g., [8,13]), like the Two Stage Maximum Likelihood (TSML) [14] algorithm discussed in Section 2. Actually, a significant loss of accuracy is experienced when some configurations of the ANs are used. Unfortunately, the most problematic configurations are those in which the ANs are (almost) coplanar, which is very common in real indoor environments because ANs are normally located near the ceiling of the environment in order to maximize coverage, to minimize multipath interference, and to limit the interference caused by people, furniture, and other obstacles. Such a very common choice degrades the performance of ordinary algorithms because some of the matrices involved in such algorithms become strongly ill conditioned (e.g., [15]). Observe that the loss of accuracy for critical arrangements of the ANs is a characteristic shared by many localization algorithms, and it is neither related to the precision of the ranging technology used to perform localization nor to the floating-point accuracy of computation. The only way to prevent such problems is the relocation of the ANs, which is often impractical in indoor scenarios because the choice of the positions of the ANs is often driven by the practical considerations mentioned previously.

The localization as optimization approach was proposed in previous works (e.g., [16,17]) with the aim of reducing the loss of accuracy that is caused by the use of ordinary algorithms when the configuration of ANs is critical. The proposed approach, as recalled in Section 2, is based on the possibility of reformulating a localization problem in terms of a related optimization problem, which is then solved using one of the algorithms documented in the vast literature on nonlinear optimization. Note that some optimization algorithms still have problems related to ill conditioned matrices, so the choice of the optimization algorithm to be used in the context of localization as optimization is crucial to avoid the loss of accuracy for critical arrangements of the ANs. The use of Particle Swarm Optimization (PSO) [18] was proposed to support localization as optimization in [12,15,17] with the so-called PSO (localization) algorithm. Unfortunately, even if the PSO algorithm proved to be effective at reducing the loss of accuracy caused by critical arrangements of ANs, PSO has the following intrinsic characteristics that make it unsatisfactory to support localization. First, it does not guarantee that global extrema are computed in a finite time. Second, its performance depends on a set of values that can be determined only by means of detailed simulations of the algorithm in the considered environment, and the performance may vary significantly when different environments are considered.

In this paper, the Polynomial Optimization using Subdivision Trees (POST) algorithmic framework is proposed as an effective means to support localization as optimization. The POST framework uses a technique inspired by the research on polynomial optimization, and it is completed with suitable methods to compute lower and upper bounds of polynomial functions to obtain concrete algorithms to solve the minimization problems derived from localization problems. In particular, in this paper, the instantiation of the POST framework that uses recent results to compute needed lower and upper bounds is presented and studied under the name of the  $POST_U$  algorithm. The  $POST_U$  algorithm, as described in Section 3, is inspired by recent results [19] on constraint satisfaction problems according to which the lower and upper bounds of a polynomial function over a box can be computed by evaluating the function at the corners of the box. In detail, once the localization problem at hand

is rewritten into a related minimization problem, the  $POST_U$  algorithm recursively subdivides the considered environment into disjoint sub-boxes to search for the minimum of the polynomial function related to the localization problem using the mentioned lower and upper bounds to appropriately prune the search space. Note that the accuracy of the  $POST_U$  algorithm does not depend on the positions of the ANs, and therefore, the  $POST_U$  algorithm can also be used when the accuracy of ordinary algorithms, like the TSML algorithm, degrades sensibly. In addition, note that the experimental results discussed in Section 4 show that the  $POST_U$  algorithm ensures an accuracy of localization analogous to the accuracy of the TSML algorithm when the arrangement of ANs is not critical for the TSML algorithm. Such a comparison with the TSML algorithm is particularly relevant because the TSML algorithm is an accepted reference to assess the performance of localization algorithms since it can attain the Cramér–Rao lower bound for the position estimator (e.g., [20]). Finally, note that the experimental results discussed in Section 4 show that the performance of the  $POST_U$  algorithm is very similar to the performance of the PSO algorithm, but the  $POST_U$  algorithm does not have the two problematic characteristics of the PSO algorithm mentioned previously.

The experimental campaign discussed in Section 4 on the use of the  $POST_U$  algorithm to estimate the position of a TN in a large empty corridor was performed using a novel localization add-on module for the Java Agent and DEvelopment framework (JADE) [21] that provides software agents with the ability to estimate the position of the devices that host them within a known environment. Note that JADE and its companion tools (e.g., [22]) are consolidated technologies for software agents, and they are promising technologies in the context of location based services (e.g., [23]) because of the advanced interaction mechanisms [24] that software agents natively provide. The presented experiments compare the  $POST_U$  algorithm with the TSML algorithm and with the PSO algorithm, and they use UWB signaling to estimate the distances from the TN to the available ANs. In the presented experiments, the ANs are fixed UWB nodes specifically installed to support localization, and four different arrangements of the ANs are given. For each arrangement, the TN is located at eight different positions inside the corridor. In the first considered scenario, all ANs are installed at the same height, close to the ceiling of the corridor. Observe that, in such a configuration, the TSML algorithm is not applicable because of the reasons mentioned previously. In the second scenario, the heights of two ANs are slightly decreased to ensure that the TSML algorithm is applicable. In this case, even if it is applicable, the results obtained using the TSML algorithm are less accurate than the results obtained with the localization as optimization approach. In the third scenario, the two ANs whose heights were decreased are further lowered to improve the accuracy of the TSML algorithm. Finally, in the last scenario, the two relocated ANs are moved close to the floor of the corridor, while the other two ANs are still installed close to the ceiling. As expected, experimental results confirm that the accuracy of the  $POST_U$  algorithm is independent of the arrangement of the ANs and that, in all experimented arrangements, its accuracy is better than the accuracy of the TSML algorithm, and it is analogous to the accuracy of the PSO algorithm. The obtained results are compatible with targeted application scenarios because the average localization error of the  $POST_U$  algorithm, measured as the distance between the true position of the TN and its estimated position, is below 1 m. Note that all experiments were performed in an empty corridor with all ANs in the line of sight. Therefore, the accuracy of localization in practical situations is expected to degrade with the presence of obstacles and multipath interference. Actually, the possibility of installing all ANs near the ceiling of the considered environment is relevant in this respect because it can significantly increase line of sight coverage.

This paper is organized as follows. Section 2 introduces the notation adopted throughout the paper, and it formulates the localization problem both from a geometric point of view and from an optimization based point of view. For the sake of completeness, Section 2 also summarizes the TSML algorithm and the PSO algorithm. Section 3 describes the proposed  $POST_U$  algorithm, and it also presents the pseudocode of a function that implements the proposed algorithm. Section 4 discusses the results of the experimental campaign performed to assess the effectiveness of the proposed algorithm. Finally, Section 5 concludes the paper and outlines future research directions.

## 2. Localization as Optimization and Reference Algorithms

In this section, the notation adopted throughout the paper is introduced, and the details of the localization as optimization approach are described. The two reference algorithms, namely the TSM algorithm and the PSO algorithm, are also briefly recalled. The  $POST_U$  algorithm is described in Section 3, and it is compared with the two reference algorithms in Section 4.

### 2.1. Localization as Optimization

In order to perform indoor localization, the considered environment was equipped with a set of  $m \geq 4$  ANs at fixed and known locations, and the position of the  $i$ th AN is denoted as  $\mathbf{a}_i \in \mathbb{R}^3$ , where  $1 \leq i \leq m$ . Note that four is the minimum number of ANs that allows localization to be performed without ambiguity. The true position of the TN is denoted as  $\mathbf{t} \in \mathbb{R}^3$ , and using this notation, the true distance between the TN and the  $i$ th AN can be computed as:

$$r_i = \|\mathbf{t} - \mathbf{a}_i\|, \quad (1)$$

where  $1 \leq i \leq m$  and  $\|\mathbf{v}\|$  denotes the Euclidean norm of  $\mathbf{v} \in \mathbb{R}^3$ . The position of the TN corresponds to the intersection of  $m$  spheres, the  $i$ th of which is centered at  $\mathbf{a}_i$  and has radius  $r_i$ . Hence, the solution of the following system of nonlinear equations:

$$\begin{cases} \|\mathbf{t} - \mathbf{a}_1\|^2 = r_1^2 \\ \|\mathbf{t} - \mathbf{a}_2\|^2 = r_2^2 \\ \vdots \\ \|\mathbf{t} - \mathbf{a}_m\|^2 = r_m^2 \end{cases} \quad (2)$$

is the true position of the TN. In practical localization scenarios, the values of distances  $\{r_i\}_{i=1}^m$  are unknown, and the position of the TN is estimated using a set of distance estimates acquired by means of a proper ranging technology. Denoting as  $\tilde{r}_i$  the estimated distance from the TN to the  $i$ th AN, the following equality holds for  $1 \leq i \leq m$ :

$$\tilde{r}_i = \|\tilde{\mathbf{t}} - \mathbf{a}_i\|, \quad (3)$$

where  $\tilde{\mathbf{t}} \in \mathbb{R}^3$  corresponds to the estimated position of the TN. Therefore, the following system of nonlinear equations:

$$\begin{cases} \|\tilde{\mathbf{t}} - \mathbf{a}_1\|^2 = \tilde{r}_1^2 \\ \|\tilde{\mathbf{t}} - \mathbf{a}_2\|^2 = \tilde{r}_2^2 \\ \vdots \\ \|\tilde{\mathbf{t}} - \mathbf{a}_m\|^2 = \tilde{r}_m^2 \end{cases} \quad (4)$$

must be satisfied by the estimated position of the TN. Note that (4) is obtained from (2) by substituting the true values with the corresponding estimated values. Furthermore, note that while the spheres represented in (2) always intersect in a single point, which corresponds to the position of the TN, (4) may not have a single solution due to the errors introduced by the measurement of distances. For this reason, proper localization algorithms are necessary to estimate the position of the TN. The large majority of localization algorithms proposed in the literature (e.g., [13]) require algebraic manipulations of matrices to solve (4), and they face severe problems when the manipulated matrices are ill conditioned. The only way to prevent such problems is to relocate the ANs, which is often impractical in indoor scenarios because the positions of the ANs in such cases is often decided on the basis of practical considerations regarding line of sight coverage.

The localization as optimization approach is based on the possibility of rewriting a localization problem into a related optimization problem. First, note that (4) can be rewritten in matrix notation as:

$$\mathbf{1} \tilde{\mathbf{t}}^T \tilde{\mathbf{t}} - 2A \tilde{\mathbf{t}} = \tilde{\mathbf{k}}, \quad (5)$$

where  $\mathbf{1} \in \mathbb{R}^m$  is the vector whose  $m$  components equal one,  $\tilde{\mathbf{k}} \in \mathbb{R}^m$  is the vector whose  $i$ th component is  $\tilde{k}_i = \tilde{r}_i^2 - \|\mathbf{a}_i\|^2$ , and  $A$  is the following  $m \times 3$  matrix, commonly called the anchor matrix:

$$A = \begin{pmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \vdots \\ \mathbf{a}_m^T \end{pmatrix}. \quad (6)$$

The solution  $\tilde{\mathbf{t}}$  of (5), which is the estimated position of the TN, can be found by solving the following minimization problem:

$$\tilde{\mathbf{t}} = \underset{\mathbf{x} \in E}{\operatorname{argmin}} \lambda(\mathbf{x}), \quad (7)$$

where  $E \subseteq \mathbb{R}^3$  is the indoor environment in which the TN is supposed to be located and  $\lambda : E \mapsto \mathbb{R}$  is the localization cost function to be minimized:

$$\lambda(\mathbf{x}) = \|\mathbf{1} \mathbf{x}^T \mathbf{x} - 2A \mathbf{x} - \tilde{\mathbf{k}}\|^2. \quad (8)$$

Note that the use of the localization as optimization approach provides a way to estimate the position of the TN even if (5) does not have a single solution because all solutions to the minimization problem (7) are considered equally valid estimates of the position of the TN.

In the remainder of this section, the two reference algorithms are briefly recalled. The first algorithm is the mentioned TSML algorithm, which was introduced in [14] to solve localization problems using the maximum likelihood approach. The second algorithm is based on PSO, and it was introduced in [25] as a first attempt to use the localization as optimization approach.

## 2.2. The TSML Algorithm

As suggested by its name, the Two Stage Maximum Likelihood (TSML) [14] algorithm is structured in two cascaded phases. In the first phase, (4) is rewritten in matrix notation as:

$$G\mathbf{v} = \tilde{\mathbf{h}}_1, \quad (9)$$

where the following definitions for matrix  $G$  and for vectors  $\mathbf{v}$  and  $\tilde{\mathbf{h}}_1$  are used to embed the characteristics of the localization problem properly:

$$G = \begin{pmatrix} -2\mathbf{a}_1^T & 1 \\ -2\mathbf{a}_2^T & 1 \\ \vdots & \vdots \\ -2\mathbf{a}_m^T & 1 \end{pmatrix} \quad \mathbf{v} = \begin{pmatrix} \tilde{\mathbf{t}} \\ \|\tilde{\mathbf{t}}\|^2 \end{pmatrix} \quad \tilde{\mathbf{h}}_1 = \begin{pmatrix} \tilde{r}_1^2 - \|\mathbf{a}_1\|^2 \\ \tilde{r}_2^2 - \|\mathbf{a}_2\|^2 \\ \vdots \\ \tilde{r}_m^2 - \|\mathbf{a}_m\|^2 \end{pmatrix}. \quad (10)$$

Note that (9) is not a linear system of equations because the last component of  $\mathbf{v}$  is related to its first three components. However, in the first phase of the algorithm, (9) is considered as a linear equation, and it is solved using the maximum likelihood approach. In order to take into account that the last component of  $\mathbf{v}$  depends on its first three components, the second phase of the TSML algorithm is needed. In the second phase of the algorithm, a second system of equations (e.g., [14]) is solved according to the maximum likelihood approach to compute the estimated position of the TN.

### 2.3. The PSO Algorithm

The large majority of localization algorithms (e.g., [13]), like the TSML algorithm, require algebraic manipulations of matrices whose entries depend on the positions of the TN and of the ANs. One of the inherent problems of such algorithms is that the matrices involved in the computations become ill conditioned for critical arrangements of the ANs. As discussed previously, in order to reduce the errors introduced by ill conditioned matrices, the localization as optimization approach was proposed (e.g., [16,17]). Such an approach is based on the possibility of rewriting a localization problem into the minimization problem posed in (7) and solving the minimization problem using one of the available optimization algorithms. Particle Swarm Optimization (PSO) [18] can be used to solve optimization problems, and hence, it can be used to support the localization as optimization approach in terms of the so-called PSO (localization) algorithm (e.g., [17]).

During the initialization phase of the PSO algorithm,  $s$  particles are positioned in the search space, which corresponds to the indoor environment in which localization is performed. The positions of the particles are randomly initialized inside the search space, and they are denoted as  $\mathbf{p}^{(i)}(0)$ , where  $1 \leq i \leq s$ . Each particle is also associated with a velocity, which is randomly initialized and which is denoted as  $\mathbf{w}^{(i)}(0)$ , where  $1 \leq i \leq s$ . After the initialization phase, the positions and the velocities of all particles are updated through subsequent iterations in order to move particles toward the position that corresponds to the solution of the considered optimization problem (e.g., [26]). The position and the velocity of the  $i$ th particle at the  $t$ th iteration are denoted as  $\mathbf{p}^{(i)}(t)$  and  $\mathbf{w}^{(i)}(t)$ , respectively. In order to describe the rule adopted to update the velocity of a generic particle, the set  $P^{(i)}(t) = \{\mathbf{p}^{(i)}(0), \dots, \mathbf{p}^{(i)}(t)\}$  is introduced to represent the set of all the positions reached by the  $i$ th particle from initialization to iteration  $t$ . The best position reached at iteration  $t$  by the  $i$ th particle can be expressed as:

$$\mathbf{y}^{(i)}(t) = \underset{\mathbf{z} \in P^{(i)}(t)}{\operatorname{argmin}} \lambda(\mathbf{z}), \quad (11)$$

where  $\lambda$  is the localization cost function introduced in (8). Similarly,  $Y(t) = \{\mathbf{y}^{(1)}(t), \dots, \mathbf{y}^{(s)}(t)\}$  is introduced to represent the set of the best positions reached by the particles from initialization to iteration  $t$ . Therefore, the global best position reached at iteration  $t$  by the particles in the swarm can be expressed as follows:

$$\mathbf{y}(t) = \underset{\mathbf{z} \in Y(t)}{\operatorname{argmin}} \lambda(\mathbf{z}), \quad (12)$$

where  $\lambda$  is the localization cost function introduced in (8). Using this notation, the velocity of the  $i$ th particle is updated according to the following rule (e.g., [26]):

$$\mathbf{w}^{(i)}(t+1) = \omega \mathbf{w}^{(i)}(t) + c_1 R_1 (\mathbf{y}^{(i)}(t) - \mathbf{p}^{(i)}(t)) + c_2 R_2 (\mathbf{y}(t) - \mathbf{p}^{(i)}(t)), \quad (13)$$

where the following quantities are introduced:

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

After updating the velocities according to (13), the position of the  $i$ th particle is updated as follows:

$$\mathbf{p}^{(i)}(t+1) = \mathbf{p}^{(i)}(t) + \mathbf{w}^{(i)}(t+1). \quad (14)$$

The PSO algorithm is iterated until a termination condition is met. One of the possible termination conditions is reaching a maximum number of iterations. At the end of the execution of the algorithm, the solution of the optimization problem is the global best position, and in the context of localization as optimization, such a solution corresponds to the estimated position of the TN. The experimental results presented in Section 4 were obtained with  $s = 40$ ,  $\omega = 0.5$ ,  $c_1 = c_2 = 2$ , and the adopted termination condition corresponds to the reach of 50 iterations.



### 3. The $\text{POST}_U$ Algorithm

As discussed in previous sections, the localization as optimization approach prevents problems related to the (possibly) ill conditioned matrices that are manipulated by well known (geometric) localization algorithms. Note that many of the algorithms for nonlinear optimization still have problems related to ill conditioned matrices, and rewriting into an optimization problem does not ensure that the loss of accuracy for critical arrangements of the ANs is avoided, unless specific attention is paid to the choice of the optimization algorithm. PSO, as briefly outlined at the end of the previous section, is one of the algorithms that can be used to support the localization as optimization approach among the plethora of optimization algorithms available in the vast literature on the subject. In this section, a novel algorithm that can be used to support the localization as optimization approach, namely the  $\text{POST}_U$  algorithm, is described and briefly studied. The notation used to describe the  $\text{POST}_U$  algorithm borrows common notations, which include the multi-index notation and a notation to describe boxes, that are normally used to study real functions of several real variables. As such, the adopted notation should be easily understandable, but interested readers can refer to [27] for a detailed description of the notations and conventions followed.

The localization cost function  $\lambda$  defined in (8) is a polynomial function of three real variables. The three variables represent the coordinates of the TN inside the considered indoor environment, and the maximum degree of each variable in  $\lambda$  is four. Hence, the localization cost function is a multivariate polynomial function of  $n = 3$  real variables with multi-degree  $L = (4, 4, 4)$ , and the localization problem is solved by finding the minimum of  $\lambda$  in the indoor environment  $E \subseteq \mathbb{R}^3$  in which the TN is supposed to be located. As discussed in Section 1, one of the assumptions of the proposed algorithm is that the indoor environment in which localization is performed can be split into boxes. For this reason, the considered minimization problem can be restricted to boxes.

The vast literature on nonlinear programming proposes several algorithms that are designed to solve problems in which the cost function is polynomial and the minimum is searched in a box. Many of such algorithms make use of one of the most relevant properties of the Bernstein polynomial basis (e.g., [28] for a comprehensive review of the subject and a historical retrospective), according to which the lower and upper bounds of a polynomial function over a box can be computed by means of the Bernstein coefficients [28] of the function. Such bounds can be used to drive the subdivision of the considered box to search for the extrema of the considered function in the box (e.g., [28]) using an ordinary branch-and-bound approach. Unfortunately, such a method based on the Bernstein polynomial basis is problematic when the polynomial function to minimize is obtained from a localization problem because the number of variables  $n = 3$  and the multi-degree of the polynomial function  $L = (4, 4, 4)$  require that 125 Bernstein coefficients are computed in the worst case for each considered sub-box. A preliminary analysis showed that, even if effective algorithms for the computation of Bernstein coefficients are used (e.g., [29]), the computational cost of the method is too high for the intended applications of indoor localization briefly mentioned in Section 1.

A related method that is expected to be less expensive in terms of the inherent computational cost can be derived from a recent result [19] that originates from the research on the satisfaction of polynomial constraints over finite domains [30]. Actually, the mentioned result can be used to compute lower and upper bounds for a polynomial function over a box without using Bernstein coefficients. Even though the obtained bounds are typically less strict than the bounds obtained using Bernstein coefficients, the use of the mentioned result is preferable because the expected computational cost for  $n = 3$  variables and for multi-degree  $L = (4, 4, 4)$  is expected to be compatible with the intended applications of indoor localization briefly discussed in Section 1.

In detail, the following proposition from [19] generalizes known results for one and two variables, and it can be used to compute lower and upper bounds for a polynomial function of  $n$  real variables over the box  $U = [0, 1]^n$ . The proof of the proposition is technical, and it is omitted for the sake of brevity. Interested readers should consult [19] for further details on the proposition and for discussions on possible extensions and generalizations.

**Proposition 1.** Given a polynomial function  $p : \mathbb{R}^n \mapsto \mathbb{R}$  of  $n \in \mathbb{N}_+$  real variables  $\mathbf{x} = (x_k)_{k=1}^n$  with multi-degree  $L \in \mathbb{N}^n$ :

$$p(\mathbf{x}) = \sum_{I \leq L} a_I \mathbf{x}^I, \quad (15)$$

if  $U = [0, 1]^n$  and:

$$\underline{p} = \min_{\mathbf{x} \in U} p(\mathbf{x}) \quad \bar{p} = \max_{\mathbf{x} \in U} p(\mathbf{x}), \quad (16)$$

then:

$$\min_{\mathbf{x} \in \dot{U}} p(\mathbf{x}) - \delta_U \leq \underline{p} \quad \bar{p} \leq \max_{\mathbf{x} \in \dot{U}} p(\mathbf{x}) + \delta_U, \quad (17)$$

where:

$$\delta_U = \frac{1}{8} \sum_{I \leq L} |I|(|I| - 1)|a_I|, \quad (18)$$

and  $\dot{U} = \{0, 1\}^n$  is the set of the corners of  $U$ .

Observe that the assumption in Proposition 1 that only box  $U$  can be considered to compute lower and upper bounds is not restrictive. Actually, in order to consider a polynomial function  $p : \mathbb{R}^n \mapsto \mathbb{R}$  of  $n \in \mathbb{N}_+$  real variables  $\mathbf{x} = (x_k)_{k=1}^n$  that take values from an arbitrary box  $B = [\underline{\mathbf{b}}, \bar{\mathbf{b}}]$ , it is sufficient to change the variables of  $p$  to  $\mathbf{u} = (u_k)_{k=1}^n$ , where:

$$\mathbf{x} = (\bar{\mathbf{b}} - \underline{\mathbf{b}})\mathbf{u} + \underline{\mathbf{b}}. \quad (19)$$

The  $\text{POST}_U$  algorithm uses the bounds computed in Proposition 1 to drive the subdivision of the initial box, which corresponds to the considered environment, to search for the minimum of the localization cost function  $\lambda$  defined in (8). In detail, the pseudocode of the  $\text{POST}_U$  function, which implements the  $\text{POST}_U$  algorithm, is shown in Algorithm 1. The parameters of the function are:

- The vector of measured distances  $\tilde{\mathbf{r}} \in \mathbb{R}^m$  from the  $m \geq 4$  visible ANs (Line 2);
- The current box  $B \subset \mathbb{R}^3$  in which the TN is supposed to be located (Line 3); and
- The current estimate  $\hat{\mathbf{t}} \in \mathbb{R}$  of the position of the TN (Line 4).

The result of the function is the estimated position  $\tilde{\mathbf{t}} \in B$  of the TN (Line 5). In order to compute the needed lower and upper bounds for the localization cost function  $\lambda$  over the current box, the  $\text{POST}_U$  function passes the anchor matrix  $A$  defined in (6), together with  $\tilde{\mathbf{r}}$  and  $B$ , to the  $\text{Bounds}_U$  function (Line 6), which actually computes the lower and upper bounds using Proposition 1. The termination condition of the  $\text{POST}_U$  function is given in terms of the parameter  $\epsilon > 0$ , which denotes the tolerance of the computed position estimate. In particular,  $\epsilon$  is used to identify when considered boxes are sufficiently small (Line 13), and it is chosen such that the localization cost function  $\lambda$  can be considered almost constant in a box whose longest edge is  $\epsilon$ . The value of  $\epsilon$  ultimately expresses the accepted tolerance on the computed position estimates. The experimental results shown in Section 4 for the  $\text{POST}_U$  algorithm were obtained using  $\epsilon = 10^{-2}$ , so that two position estimates whose coordinates differ by less than 1 cm are considered equal. Such a choice is reasonable, especially in view of the errors introduced by the underlying ranging technology. Actually, different values of  $\epsilon$  can be chosen, depending on the targeted application and on the adopted ranging technology.

The  $\text{POST}_U$  function is computed by a software agent hosted on a mobile device to estimate the position of the device within the considered environment as soon as the distance estimates to  $m \geq 4$  ANs become available. The arguments passed to the function are: the acquired distance estimates  $\tilde{\mathbf{r}} \in \mathbb{R}^m$ , the box that describes the entire considered environment, and the coordinates of a point inside the box. Note that such arguments could be improved if the device could track its movements, so that it could provide the agent with a good initial estimate of its position. Such a possibility, which is feasible for mobile devices like smartphones because they are readily equipped with needed sensors, is not considered in this work.



**Algorithm 1** Pseudocode of the  $\text{POST}_U$  function, which is used to estimate the position  $\tilde{\mathbf{t}} \in B$  of the TN in box  $B = [\underline{\mathbf{b}}, \overline{\mathbf{b}}] \subset \mathbb{R}^3$  using estimated distances  $\tilde{\mathbf{r}} \in \mathbb{R}^m$  from  $m \geq 4$  ANs together with an initial approximation  $\hat{\mathbf{t}}$  of the position of the TN. The  $m \times 3$  anchor matrix  $A$  contains the positions of the ANs, and  $\epsilon \in \mathbb{R}_+$  is the accepted tolerance.

---

```

1: function  $\text{POST}_U(\tilde{\mathbf{r}}, B, \hat{\mathbf{t}})$  ▷ the  $\text{POST}_U$  function
2: input  $\tilde{\mathbf{r}} \in \mathbb{R}^m$  ▷ distances from the  $m \geq 4$  ANs
3: input  $B \subset \mathbb{R}^3$  ▷ the box in which the TN is supposed to be located
4: input  $\hat{\mathbf{t}} \in B$  ▷ the current estimate of the position of the TN
5: output  $\tilde{\mathbf{t}} \in B$  ▷ an estimate of the position of the TN
6:  $(l, u) \leftarrow \text{Bounds}_U(\tilde{\mathbf{r}}, B, A)$  ▷ compute the lower and upper bounds of  $\lambda$ 
7: if  $l > \lambda(\hat{\mathbf{t}})$  then ▷ is  $l$  less than the current minimum of  $\lambda$ ?
8:   return  $\hat{\mathbf{t}}$  ▷ the current minimum of  $\lambda$  does not decrease in  $B$ 
9: end if
10:  $b \leftarrow \max\{\bar{b}_1 - \underline{b}_1, \bar{b}_2 - \underline{b}_2, \bar{b}_3 - \underline{b}_3\}$  ▷  $b$  is the length of the longest edge of  $B$ 
11: if  $b < \epsilon$  then ▷ is  $B$  sufficiently small?
12:   select  $\tilde{\mathbf{t}} \in B$  ▷ select a point in  $B$ 
13:   if  $u < \lambda(\hat{\mathbf{t}}) \wedge \lambda(\tilde{\mathbf{t}}) < \lambda(\hat{\mathbf{t}})$  then ▷ does  $\tilde{\mathbf{t}}$  improve the current minimum of  $\lambda$ ?
14:     return  $\tilde{\mathbf{t}}$  ▷  $\tilde{\mathbf{t}}$  improves the current minimum of  $\lambda$ 
15:   else
16:     return  $\hat{\mathbf{t}}$  ▷  $\tilde{\mathbf{t}}$  does not improve the current minimum of  $\lambda$ 
17:   end if
18: end if
19: select  $1 \leq i \leq 3$  ▷ select an axis to split  $B$ 
20: select  $s \in [\underline{b}_i, \bar{b}_i]$  ▷ select a point in  $B = [\underline{\mathbf{b}}, \overline{\mathbf{b}}]$  along the chosen axis
21:  $I \leftarrow [\underline{b}_i, s]$  ▷ split  $B$  along the chosen axis at  $s$ , and consider the left sub-box
22:  $\tilde{\mathbf{t}}_l \leftarrow \text{POST}_U(\tilde{\mathbf{r}}, B_{i \rightarrow l}, \hat{\mathbf{t}})$  ▷ recurse to improve the minimum of  $\lambda$  possibly
23: if  $\lambda(\tilde{\mathbf{t}}_l) < \lambda(\hat{\mathbf{t}})$  then ▷ does the left sub-box provide an improved minimum of  $\lambda$ ?
24:    $\hat{\mathbf{t}} \leftarrow \tilde{\mathbf{t}}_l$  ▷ the minimum is improved in the left sub-box
25: else
26:    $\hat{\mathbf{t}} \leftarrow \hat{\mathbf{t}}$  ▷ the minimum is not improved in the left sub-box
27: end if
28:  $I \leftarrow [s, \bar{b}_i]$  ▷ split  $B$  along the chosen axis at  $s$ , and consider the right sub-box
29:  $\tilde{\mathbf{t}}_r \leftarrow \text{POST}_U(\tilde{\mathbf{r}}, B_{i \rightarrow r}, \hat{\mathbf{t}})$  ▷ recurse to possibly improve the minimum of  $\lambda$ 
30: if  $\lambda(\tilde{\mathbf{t}}_r) < \lambda(\hat{\mathbf{t}})$  then ▷ does the right sub-box provide an improved minimum of  $\lambda$ ?
31:    $\hat{\mathbf{t}} \leftarrow \tilde{\mathbf{t}}_r$  ▷ the minimum of  $\lambda$  is improved in the right sub-box
32: end if
33: return  $\hat{\mathbf{t}}$  ▷ return the estimated position  $\tilde{\mathbf{t}}$  of the TN
34: end function

```

---

**Proposition 2.** The  $\text{POST}_U$  function always terminates.

**Proof.** The  $\text{POST}_U$  function is recursive, and the base cases of recursion are tested at Lines 7 and 13. The recursion is terminated when the considered box cannot improve the current minimum of the location cost function (Line 7) or when the function is almost constant in the considered box, which implies that its value can be directly compared with the current minimum (Line 13). In all other cases, the recursion continues with smaller boxes computed at Lines 21 and 28. The worst case occurs when the condition at Line 7 is never satisfied, so that the reduction of the sizes of boxes would eventually produce boxes in which the localization cost function is almost constant. The proposition is proven because when the localization cost function is almost constant in a box, the recursion is immediately terminated either by accepting or by rejecting the box.  $\square$

The proof of Proposition 2 suggests a way to estimate the computational cost of the  $\text{POST}_U$  function. The following proposition estimates the computational cost of the  $\text{POST}_U$  function in terms of the number of applications of the  $\text{Bounds}_U$  function.

**Proposition 3.** *Given a box  $B = [\underline{\mathbf{b}}, \overline{\mathbf{b}}] \subset \mathbb{R}^3$ , the  $\text{POST}_U$  function terminates, in the worst case, after performing  $\mathcal{O}((\frac{b}{\epsilon})^3)$  applications of the  $\text{Bounds}_U$  function, where:*

$$b = \max\{\bar{b}_1 - \underline{b}_1, \bar{b}_2 - \underline{b}_2, \bar{b}_3 - \underline{b}_3\}. \quad (20)$$

**Proof.** Since, for Proposition 2, the  $\text{POST}_U$  function always terminates, it decomposes the initial box  $B$  into a finite number of disjoint sub-boxes. In the worst case, which corresponds to the case in which the condition at Line 7 is never satisfied, the number of processed sub-boxes is  $\mathcal{O}((\frac{b}{\epsilon})^3)$ . The proposition is proven because the  $\text{POST}_U$  function uses the  $\text{Bounds}_U$  function only at Line 6 and therefore only once for each processed sub-box.  $\square$

It is worth noting that the computational cost estimated in Proposition 3 typically overestimates the actual cost of the  $\text{POST}_U$  function in real scenarios because the condition at Line 7 is expected to prune the search tree for large sub-boxes. In addition, note that the  $\text{POST}_U$  function is open to several optimizations, which are typical of methods based on the branch-and-bound approach. In particular, the  $\text{POST}_U$  function does not detail two important choices that can be used to improve its performance in practical cases. First, the  $\text{POST}_U$  function does not specify how the axis to use for the next split of the current box is selected. Second, once the axis to use for the next split is selected, the  $\text{POST}_U$  function does not specify how a value in the current box along the selected axis is chosen. Both choices can be fixed statically, or they can be postponed at execution time to make them dynamic. In both cases, they have the potential to improve the performance of the proposed algorithm significantly in real situations, as the practice of nonlinear optimization suggests. In the current implementation, which was used for the experimental campaign discussed in Section 4, the axis to use for the next split is selected in lexicographic order, and then, the selected edge is split in the middle. Note that, even if the current implementation adopts such simple techniques, the experimental campaign showed that the proposed algorithm can be used to support real-time localization because position estimates were computed in less than 0.5 s, which is an acceptable bound for the considered applications.

#### 4. Experimental Results

This section presents experimental results that were obtained using the three algorithms described in the previous sections, namely the TSML algorithm, the PSO algorithm, and the  $\text{POST}_U$  algorithm. The three algorithms were applied offline to the same sets of distance estimates in order to make a fair comparison among them. Actually, the performance of a localization algorithm largely depended on the accuracy of available distance estimates, and a fair comparison required that the three considered algorithms were provided with the same sets of distance estimates. The TSML algorithm and the PSO algorithm were chosen as reference algorithms for the assessment of the performance of the  $\text{POST}_U$  algorithm. The TSML algorithm was chosen because it can attain the Cramér–Rao lower bound for the position estimator (e.g., [20]). The PSO algorithm was chosen because it is based on the same approach adopted by the  $\text{POST}_U$  algorithm, namely the localization as optimization approach.

The experimental results discussed in this section were obtained using a commercial Android smartphone called SpoonPhone ([www.bespoon.com](http://www.bespoon.com)). To the best of our knowledge, the SpoonPhone is the only commercial smartphone on the market that allows measuring distances using UWB signaling through special hardware and software modules. Such modules can be used to estimate the distances to coupled active tags by means of a dedicated interface provided by the manufacturer. Then, acquired distance estimates can be used to feed the considered localization algorithms. All the experiments were performed using a SpoonPhone as TN and four active tags as ANs. The indoor environment in which the experiments were performed was a 4 m wide, 10 m long, and 3 m tall corridor. It is worth noting that the corridor was empty during the distance acquisition phases of the experiments in order to guarantee line of sight visibility and to minimize distance errors caused by obstacles and multipath interference. An experimental campaign to assess the performance of the proposed algorithm when the environment contains obstacles and people is planned for future work.

Four arrangements of the ANs that differed only for the heights of two of the four ANs were considered in the experiments. For each arrangement of the ANs, the TN was placed at eight different locations in the considered environment. The experimented positions of the TN were the same for the four arrangements of the ANs. The considered scenarios are schematically shown in Figure 1, where the positions of the TN are marked in red and numbered from 1 to 8. The coordinates of the four positions of the TN numbered from 1 to 4 are expressed in meters in the coordinate system shown in Figure 1 as:

$$\mathbf{t}_1 = (0, 0, 1.5) \quad \mathbf{t}_2 = (2, 0, 1.5) \quad \mathbf{t}_3 = (2, 5, 1.5) \quad \mathbf{t}_4 = (0, 5, 1.5). \quad (21)$$

Note that the height of such positions is compatible with the height of a smartphone used for augmented reality applications because the user, which is a kid in a typical usage scenario, holds the smartphone up to see through it. The coordinates of the four positions of the TN numbered from 5 to 8 are expressed in meters in the considered coordinate system as:

$$\mathbf{t}_5 = (0, 0, 1) \quad \mathbf{t}_6 = (2, 0, 1) \quad \mathbf{t}_7 = (2, 5, 1) \quad \mathbf{t}_8 = (0, 5, 1). \quad (22)$$

Note that the abscissa and ordinate were the same as before, while the height was lowered to 1 m to make it compatible with the height of a smartphone used for applications, like educational games for kids in exhibitions and museums, that require the user to navigate the environment using a virtual compass.

The performance of the three considered algorithms was evaluated, for the eight positions of the TN and for the four arrangements of the ANs, in terms of the measured localization error. In detail, using the nomenclature introduced in Section 2, the localization error measured to compare the three algorithms can be defined as:

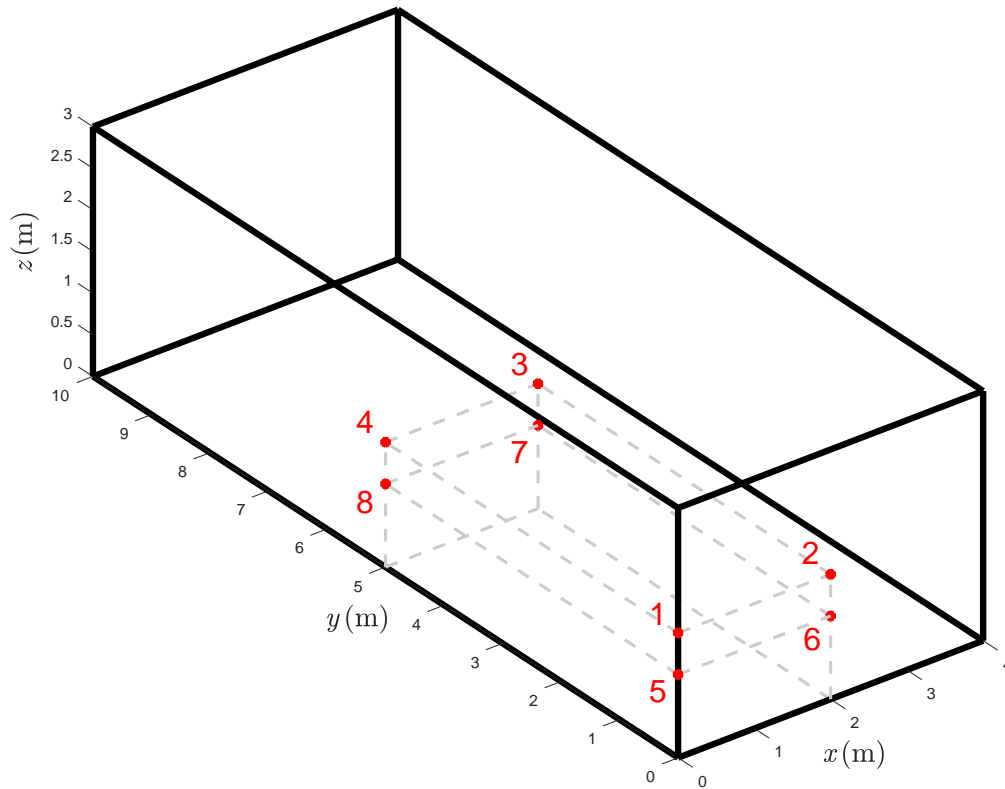
$$\mathbf{e} = \mathbf{t} - \tilde{\mathbf{t}}, \quad (23)$$

where  $\mathbf{t}$  denotes the true position of the TN and  $\tilde{\mathbf{t}}$  denotes its estimated position.

For each one of the  $c = 32$  experimental configurations, the distances from the TN to the ANs were acquired  $r = 100$  times, and the following quantities were computed for each considered algorithm:

- The average and the standard deviation of the absolute value of the first component of  $\mathbf{e}$  over the  $r$  acquisitions obtained using the TSML algorithm, denoted as  $e_T^x$  and  $\sigma_T^x$ , the analogous values obtained using the PSO algorithm, denoted as  $e_P^x$  and  $\sigma_P^x$ , and the analogous values obtained using the POST<sub>U</sub> algorithm, denoted as  $e_U^x$  and  $\sigma_U^x$ ;
- The average and the standard deviation of the absolute value of the second component of  $\mathbf{e}$  over the  $r$  acquisitions obtained using the TSML algorithm, denoted as  $e_T^y$  and  $\sigma_T^y$ , the analogous values obtained using the PSO algorithm, denoted as  $e_P^y$  and  $\sigma_P^y$ , and the analogous values obtained using the POST<sub>U</sub> algorithm, denoted as  $e_U^y$  and  $\sigma_U^y$ ; and
- The average and the standard deviation of the Euclidean norm of the vector obtained with the first two components of  $\mathbf{e}$  over the  $r$  acquisitions obtained using the TSML algorithm, denoted as  $e_T$  and  $\sigma_T$ , the analogous values obtained using the PSO algorithm, denoted as  $e_P$  and  $\sigma_P$ , and the analogous values obtained using the POST<sub>U</sub> algorithm, denoted as  $e_U$  and  $\sigma_U$ .

The most relevant reason to consider only the first two components of  $\mathbf{e}$  is that, in the envisaged applications, which include educational games for kids in exhibitions and museums, the major interest is in measuring the accuracy of the position estimates with respect to the floor of the considered indoor environment. Therefore, the errors in the estimated height of the TN were neglected in the discussed performance analysis, and only the errors in the positions with respect to the floor were considered. In the remainder of this section, the localization errors relative to the four considered localization scenarios are shown and discussed for the three considered algorithms. For each scenario, a brief comparison among the three considered algorithms is also presented in terms of the average errors and of the standard deviations mentioned previously.



**Figure 1.** The eight positions of the Target Node (TN) used for experiments are numbered and shown in red in the considered indoor environment, which is a 4 m wide, 10 m long, and 3 m tall empty corridor.

#### 4.1. Scenario 1

In the first scenario, all ANs were placed near the ceiling of the corridor schematically shown in Figure 1. In the coordinate system shown in the figure, the positions of the ANs had the following coordinates expressed in meters:

$$\mathbf{a}_1 = (0, 0, 3) \quad \mathbf{a}_2 = (4, 0, 3) \quad \mathbf{a}_3 = (4, 10, 3) \quad \mathbf{a}_4 = (0, 10, 3). \quad (24)$$

Since all the ANs shared the same height, ordinary (geometric) localization algorithms, such as the TSML algorithm, were not applicable, as discussed in Section 1. Actually, in this scenario, all the entries in the third column of matrix  $G$  in (9) were equal, and therefore, matrix  $G$  was not full rank. The TSML algorithm cannot advance to the second phase because  $\mathbf{v}$  in (9) cannot be computed.

For the PSO algorithm, Table 1 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 1.** Measured accuracy of the PSO algorithm in Scenario 1.

Position	$e_P$ (m)	$\sigma_P$ (m)	$e_P^x$ (m)	$\sigma_P^x$ (m)	$e_P^y$ (m)	$\sigma_P^y$ (m)
1	0.301	0.131	0.166	0.160	0.228	0.050
2	0.310	0.180	0.142	0.224	0.233	0.063
3	0.146	0.221	0.132	0.196	0.051	0.108
4	0.636	0.608	0.588	0.573	0.220	0.226
5	0.263	0.151	0.111	0.183	0.211	0.042
6	0.251	0.068	0.078	0.089	0.229	0.037
7	0.160	0.254	0.144	0.239	0.057	0.096
8	0.442	0.594	0.406	0.554	0.159	0.225

The lower value of  $e_P$  was 0.146 m, and the higher value was 0.636 m, which ensured an accuracy of localization compatible with the intended applications discussed in Section 1. Moreover, the lower value of the standard deviation  $\sigma_P$  was 0.068 m, and the higher value was 0.608 m. Such values of  $e_P$  and  $\sigma_P$  guaranteed accurate position estimates for all the considered positions of the TN. The values of  $e_P^x$  and  $\sigma_P^x$  were similar to the corresponding values of  $e_P^y$  and  $\sigma_P^y$ , which ensured that the position estimates were spread all around the true position of the TN and that they were not distributed along special directions.

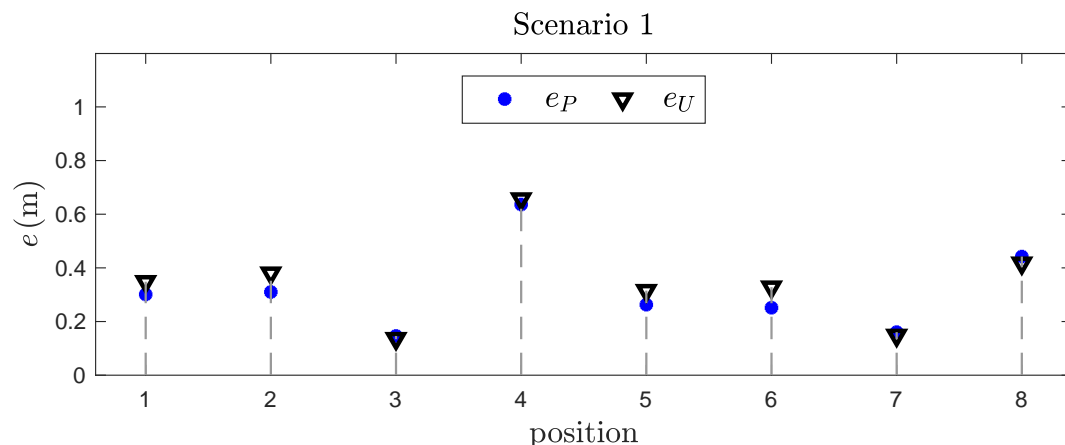
For the case of the  $POST_U$  algorithm, Table 2 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 2.** Measured accuracy of the  $POST_U$  algorithm in Scenario 1.

Position	$e_U$ (m)	$\sigma_U$ (m)	$e_U^x$ (m)	$\sigma_U^x$ (m)	$e_U^y$ (m)	$\sigma_U^y$ (m)
1	0.348	0.106	0.191	0.106	0.280	0.082
2	0.380	0.167	0.157	0.235	0.288	0.098
3	0.136	0.196	0.132	0.194	0.016	0.040
4	0.657	0.695	0.601	0.668	0.223	0.240
5	0.315	0.119	0.168	0.120	0.252	0.085
6	0.328	0.076	0.118	0.102	0.292	0.061
7	0.148	0.236	0.140	0.236	0.025	0.044
8	0.417	0.671	0.368	0.637	0.156	0.245

The lower value of  $e_U$  was 0.136 m, and the higher value was 0.657 m, which ensured a high level of localization accuracy. Table 2 also shows that the lower value of the standard deviation  $\sigma_U$  was 0.076 m and the higher value was 0.695 m. Such values of  $e_U$  and  $\sigma_U$  guaranteed a localization accuracy compatible with the envisaged applications, as discussed in Section 1. The values of  $e_U^x$  and  $\sigma_U^x$  were similar to the corresponding values of  $e_U^y$  and  $\sigma_U^y$ , which ensured that the position estimates were spread all around the true position of the TN and that the position estimates were not distributed along special directions. Observe that the values of the average localization errors and of standard deviations of localization errors for the  $POST_U$  algorithm were similar to the analogous values obtained with the PSO algorithm.

Figure 2 shows a summary of the experimental results for the first scenario. Only the average errors of the position estimates with respect to the floor of the considered environment are shown because such values were considered the most relevant quantities to assess the performance of localization algorithms in the envisaged applications. Note that the experimental results for the TSM algorithm are not shown because such an algorithm is not usable in this scenario.



**Figure 2.** The average errors in Scenario 1 for the PSO algorithm ( $e_P$ ) and for the  $POST_U$  algorithm ( $e_U$ ) shown in meters.

#### 4.2. Scenario 2

In order to obtain experimental results for the TSML algorithm, two of the ANs were lowered with respect to the first scenario. Even if no obstacles were present during the experimental campaign, the heights of the two relocated ANs were intentionally kept close to the ceiling of the corridor to reduce the effects of obstacles in practical applications. In particular, the relocated ANs were those numbered 1 and 3, and the positions of the ANs in this scenario had the following coordinates expressed in meters in the coordinate system shown in Figure 1:

$$\mathbf{a}_1 = (0, 0, 2.5) \quad \mathbf{a}_2 = (4, 0, 3) \quad \mathbf{a}_3 = (4, 10, 2.5) \quad \mathbf{a}_4 = (0, 10, 3). \quad (25)$$

The TSML algorithm was applicable in this scenario, and Table 3 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 3.** Measured accuracy of the Two Stage Maximum Likelihood (TSML) algorithm in Scenario 2.

Position	$e_T$ (m)	$\sigma_T$ (m)	$e_T^x$ (m)	$\sigma_T^x$ (m)	$e_T^y$ (m)	$\sigma_T^y$ (m)
1	0.987	1.379	0.695	0.960	0.696	0.993
2	0.488	0.317	0.066	0.098	0.481	0.305
3	0.768	0.656	0.655	0.458	0.332	0.521
4	0.553	1.014	0.546	0.968	0.062	0.311
5	1.082	1.341	0.759	0.934	0.767	0.965
6	1.153	1.867	0.291	0.912	1.088	1.648
7	0.176	0.244	0.146	0.192	0.079	0.161
8	2.121	2.056	1.942	1.795	0.725	1.099

Note that the lower value of  $e_T$  was 0.176 m, and the higher value was 2.121 m; also note that the average localization error was often higher than 0.5 m. With respect to the standard deviation  $\sigma_T$ , Table 3 shows that its lower value was 0.244 m and its higher value was 2.056 m. Such values of  $e_T$  and  $\sigma_T$  did not guarantee that reliable position estimates were computed for all the considered positions of the TN. Such results were expected because matrix  $G$  in (9) was strongly ill conditioned, and therefore, the computation of  $\mathbf{v}$  from (9) was unreliable. Small errors in the measurement of distance estimates, which were reflected in the small errors in  $\tilde{\mathbf{h}}_1$  in (9), could cause large errors in the computation of  $\mathbf{v}$ . Note that such a problem was neither related to the precision of the ranging technology nor to the floating-point accuracy of computation, but it was exclusively related to the arrangement of the ANs, as discussed in Section 1.

When the PSO algorithm was used to perform localization, Table 4 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 4.** Measured accuracy of the PSO algorithm in Scenario 2.

Position	$e_P$ (m)	$\sigma_P$ (m)	$e_P^x$ (m)	$\sigma_P^x$ (m)	$e_P^y$ (m)	$\sigma_P^y$ (m)
1	0.302	0.117	0.164	0.147	0.231	0.055
2	0.261	0.062	0.081	0.072	0.242	0.043
3	0.253	0.375	0.234	0.350	0.091	0.141
4	0.113	0.220	0.101	0.217	0.033	0.053
5	0.300	0.155	0.154	0.186	0.233	0.040
6	0.335	0.287	0.172	0.337	0.221	0.054
7	0.100	0.131	0.091	0.125	0.031	0.047
8	0.496	0.554	0.450	0.524	0.178	0.211



The lower value of  $e_P$  was 0.100 m, and the higher value was 0.496 m, which ensured an accuracy of localization compatible with the intended applications discussed in Section 1. Moreover, the lower value of the standard deviation  $\sigma_P$  was 0.062 m, and the higher value was 0.554 m. Such values of  $e_P$  and  $\sigma_P$  guaranteed accurate position estimates. The values of  $e_P^x$  and  $\sigma_P^x$  were similar to the corresponding values of  $e_P^y$  and  $\sigma_P^y$ , which ensured that the position estimates were spread all around the true position of the TN and that the position estimates were distributed along special directions.

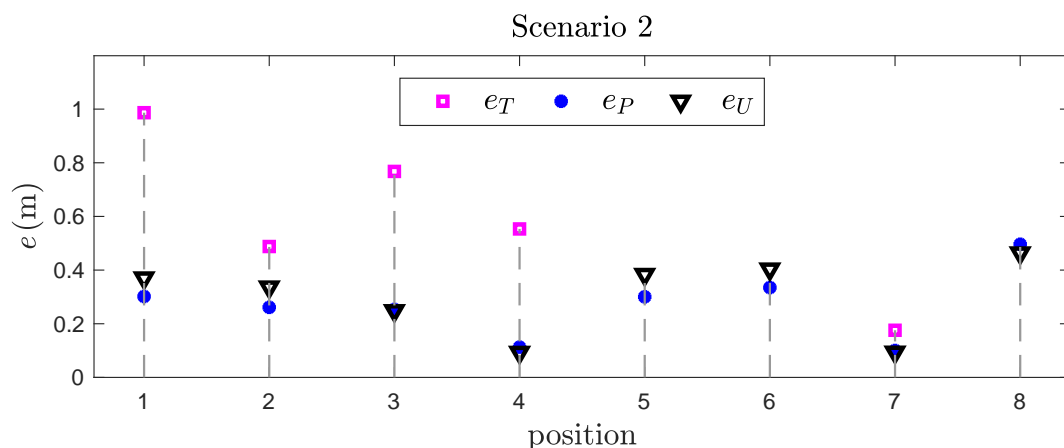
For the  $POST_U$  algorithm, Table 5 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 5.** Measured accuracy of the  $POST_U$  algorithm in Scenario 2.

Position	$e_U$ (m)	$\sigma_U$ (m)	$e_U^x$ (m)	$\sigma_U^x$ (m)	$e_U^y$ (m)	$\sigma_U^y$ (m)
1	0.370	0.109	0.212	0.117	0.291	0.077
2	0.337	0.063	0.079	0.076	0.321	0.049
3	0.248	0.368	0.242	0.363	0.029	0.076
4	0.093	0.212	0.073	0.210	0.033	0.055
5	0.384	0.126	0.231	0.129	0.293	0.083
6	0.404	0.272	0.203	0.341	0.261	0.107
7	0.094	0.128	0.089	0.121	0.021	0.047
8	0.464	0.647	0.403	0.625	0.160	0.234

The lower value of  $e_U$  was 0.093 m, and the higher value was 0.464 m, which ensured a high level of localization accuracy. Table 5 also shows that the lower value of the standard deviation  $\sigma_U$  was 0.063 m and the higher value was 0.647 m. Such values of  $e_U$  and  $\sigma_U$  guaranteed a localization accuracy compatible with the envisaged applications, as discussed in Section 1. The values of  $e_U^x$  and  $\sigma_U^x$  were similar to the corresponding values of  $e_U^y$  and  $\sigma_U^y$ , which meant that the position estimates were spread all around the true position of the TN and that the position estimates were not distributed along special directions.

Figure 3 shows a summary of the experimental results for the second scenario. In this scenario, the relocated ANs were kept sufficiently close to the ceiling of the corridor to ensure a good line of sight coverage when obstacles were present in the environment. However, even if the TSML algorithm could be used to compute estimates of the position of the TN, such an algorithm did not provide the same level of accuracy that the PSO algorithm and the  $POST_U$  algorithm ensured. Actually, the values of the adopted metrics relative to the TSML algorithm, which are shown in Table 3, were much higher than the values relative to the PSO algorithm and to the  $POST_U$  algorithm, which are shown in Table 4 and in Table 5, respectively.



**Figure 3.** The average errors in Scenario 2 for the TSML algorithm ( $e_T$ ), the PSO algorithm ( $e_P$ ), and the  $POST_U$  algorithm ( $e_U$ ) shown in meters.

Finally, note that the results obtained with the  $POST_U$  algorithm were similar to the corresponding results obtained with the PSO algorithm, as in the first scenario. Such a similarity was not surprising because the two algorithms were based on the localization as optimization approach, which ensured that the results of localization were independent of the positions of the ANs, provided that appropriate optimization algorithms were used.

### 4.3. Scenario 3

In this scenario, the height of the two relocated ANs was further reduced. The positions of the ANs in this scenario had the following coordinates expressed in meters with respect to the coordinate system shown in Figure 1:

$$\mathbf{a}_1 = (0, 0, 1.5) \quad \mathbf{a}_2 = (4, 0, 3) \quad \mathbf{a}_3 = (4, 10, 1.5) \quad \mathbf{a}_4 = (0, 10, 3). \quad (26)$$

Note that the two relocated ANs were likely to be covered by obstacles in real scenarios, which could make the estimates of the distances from such ANs unreliable. A future development of the presented research concerns the study of the characteristics of the considered algorithms in environments that contain obstacles and people.

For the TSML algorithm, Table 6 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 6.** Measured accuracy of the TSML algorithm in Scenario 3.

Position	$e_T$ (m)	$\sigma_T$ (m)	$e_T^x$ (m)	$\sigma_T^x$ (m)	$e_T^y$ (m)	$\sigma_T^y$ (m)
1	0.308	0.481	0.164	0.306	0.250	0.379
2	0.373	0.491	0.111	0.109	0.351	0.482
3	0.309	0.249	0.263	0.230	0.128	0.138
4	0.619	0.100	0.618	0.098	0.031	0.031
5	0.350	0.340	0.224	0.197	0.253	0.292
6	0.461	0.525	0.085	0.081	0.452	0.520
7	0.380	0.482	0.350	0.477	0.109	0.124
8	0.639	0.419	0.611	0.423	0.105	0.145

The lower value of  $e_T$  was 0.308 m, and the higher value was 0.639 m, which ensured an accuracy of localization compatible with the intended applications discussed in Section 1. With respect to the standard deviation  $\sigma_T$ , Table 6 shows that its lower value was 0.100 m and its higher value was 0.525 m. Such values of  $e_T$  and  $\sigma_T$  guaranteed accurate position estimates. The values of  $e_T^x$  and  $\sigma_T^x$  were similar to the corresponding values of  $e_T^y$  and  $\sigma_T^y$ , which guaranteed that the position estimates were spread all around the true position of the TN and that the position estimates were not distributed along special directions. A comparison between Table 6 and Table 3 revealed that the accuracy of the TSML algorithm in this scenario was significantly improved with respect to the accuracy obtained in the second scenario. Such a result was not surprising because the difference between the height of the two relocated ANs and the height of the other two ANs was sufficiently large.

For the PSO algorithm, Table 7 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions. Note that the lower value of  $e_P$  was 0.180 m, and the higher value was 0.331 m, which ensured an accuracy of localization compatible with the intended applications discussed in Section 1. Moreover, the lower value of the standard deviation  $\sigma_P$  was 0.091 m, and the higher value was 0.366 m. Such values of  $e_P$  and  $\sigma_P$  guaranteed accurate position estimates. In addition, the values of  $e_P^x$  and  $\sigma_P^x$  were similar to the corresponding values of  $e_P^y$  and  $\sigma_P^y$ , which ensured that the position estimates were spread all around the true position of the TN and that the position estimates were not distributed along special directions.

**Table 7.** Measured accuracy of the PSO algorithm in Scenario 3.

Position	$e_P$ (m)	$\sigma_P$ (m)	$e_P^x$ (m)	$\sigma_P^x$ (m)	$e_P^y$ (m)	$\sigma_P^y$ (m)
1	0.300	0.091	0.164	0.121	0.231	0.058
2	0.331	0.181	0.156	0.241	0.232	0.079
3	0.203	0.315	0.184	0.297	0.074	0.114
4	0.180	0.062	0.174	0.058	0.032	0.040
5	0.302	0.105	0.147	0.138	0.241	0.056
6	0.313	0.187	0.157	0.240	0.211	0.077
7	0.218	0.366	0.194	0.343	0.078	0.144
8	0.307	0.291	0.288	0.274	0.085	0.118

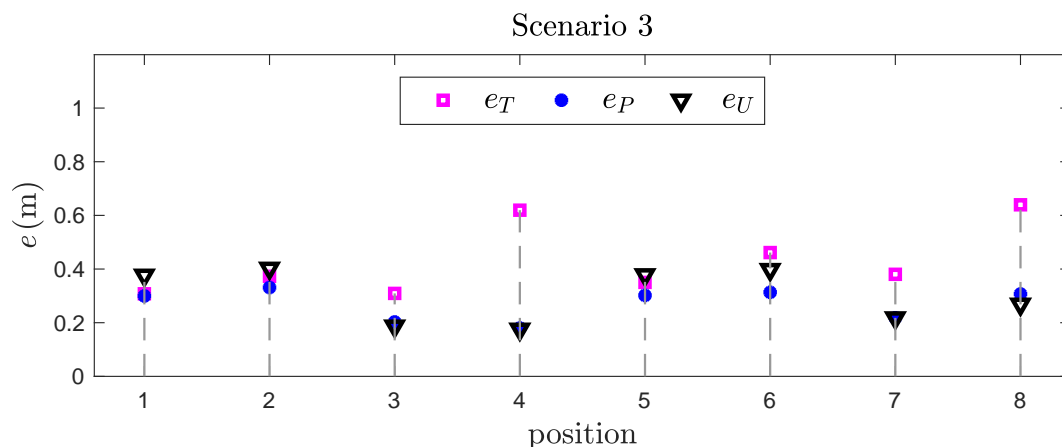
Finally, when the  $POST_U$  algorithm was used to perform localization, Table 8 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 8.** Measured accuracy of the  $POST_U$  algorithm in Scenario 3.

Position	$e_U$ (m)	$\sigma_U$ (m)	$e_U^x$ (m)	$\sigma_U^x$ (m)	$e_U^y$ (m)	$\sigma_U^y$ (m)
1	0.377	0.114	0.201	0.129	0.303	0.077
2	0.403	0.168	0.174	0.238	0.298	0.121
3	0.187	0.292	0.181	0.286	0.027	0.070
4	0.176	0.046	0.165	0.052	0.033	0.043
5	0.378	0.116	0.199	0.138	0.303	0.077
6	0.398	0.172	0.183	0.252	0.282	0.105
7	0.218	0.365	0.194	0.342	0.064	0.149
8	0.269	0.240	0.238	0.243	0.052	0.107

The lower value of  $e_U$  was 0.176 m, and the higher value was 0.403 m, which ensured a high level of localization accuracy. Table 8 also shows that the lower value of the standard deviation  $\sigma_U$  was 0.046 m, and the higher value was 0.365 m. Such values of  $e_U$  and  $\sigma_U$  guaranteed a localization accuracy compatible with the envisaged applications. The values of  $e_U^x$  and  $\sigma_U^x$  were similar to the corresponding values of  $e_U^y$  and  $\sigma_U^y$ , which ensured that the position estimates were spread all around the true position of the TN and that the position estimates were not distributed along special directions.

Figure 4 shows a summary of the experimental results for the third scenario. Note that the values obtained with the three localization algorithms were all similar in this scenario. Therefore, in the idealized case of an empty environment, the three considered algorithms achieved similar levels of accuracy, which were all sufficient to support the envisaged applications of indoor localization.



**Figure 4.** The average errors in Scenario 3 for the TSML algorithm ( $e_T$ ), the PSO algorithm ( $e_P$ ), and the  $POST_U$  algorithm ( $e_U$ ) are shown in meters.

#### 4.4. Scenario 4

In this scenario, the two relocated ANs were placed on the floor of the corridor, while the other two ANs were still close to the ceiling. With respect to the coordinate system shown in Figure 1, the positions of the ANs in this scenario had the following coordinates expressed in meters:

$$\mathbf{a}_1 = (0, 0, 0) \quad \mathbf{a}_2 = (4, 0, 3) \quad \mathbf{a}_3 = (4, 10, 0) \quad \mathbf{a}_4 = (0, 10, 3). \quad (27)$$

Note that in real applications, the two ANs positioned near the floor of the corridor were probably covered by people, furniture, and other obstacles. Therefore, in such cases, the estimates of the distances from such ANs were probably affected by large errors, which ultimately degraded the accuracy of localization, independent of the adopted localization algorithm.

For the TSML algorithm, Table 9 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 9.** Measured accuracy of the TSML algorithm in Scenario 4.

Position	$e_T$ (m)	$\sigma_T$ (m)	$e_T^x$ (m)	$\sigma_T^x$ (m)	$e_T^y$ (m)	$\sigma_T^y$ (m)
1	0.551	0.339	0.396	0.175	0.376	0.299
2	0.289	0.042	0.101	0.070	0.264	0.019
3	0.163	0.143	0.148	0.137	0.052	0.058
4	1.015	0.497	0.986	0.463	0.193	0.230
5	0.345	0.300	0.235	0.144	0.236	0.279
6	0.477	0.486	0.217	0.340	0.408	0.366
7	0.270	0.284	0.244	0.244	0.100	0.155
8	0.711	0.156	0.704	0.147	0.078	0.078

The lower value of  $e_T$  was 0.163 m, and the higher value was 1.015 m, which ensured an accuracy of localization compatible with the envisaged applications. Considering the standard deviation  $\sigma_T$ , Table 9 shows that its lower value was 0.042 m and its higher value was 0.497 m. Such values of  $e_T$  and  $\sigma_T$  guaranteed accurate position estimates. The values of  $e_T^x$  and  $\sigma_T^x$  were similar to the corresponding values of  $e_T^y$  and  $\sigma_T^y$ , which ensured that the position estimates were not distributed along special directions.

Table 10 shows the experimental results obtained using the PSO algorithm, which ensured that the obtained level of accuracy was sufficient for the envisaged applications of indoor localization.

**Table 10.** Measured accuracy of the PSO algorithm in Scenario 4.

Position	$e_P$ (m)	$\sigma_P$ (m)	$e_P^x$ (m)	$\sigma_P^x$ (m)	$e_P^y$ (m)	$\sigma_P^y$ (m)
1	0.299	0.077	0.158	0.107	0.235	0.061
2	0.276	0.045	0.076	0.060	0.261	0.031
3	0.107	0.170	0.090	0.157	0.046	0.075
4	0.503	0.589	0.458	0.563	0.154	0.224
5	0.303	0.086	0.166	0.114	0.235	0.057
6	0.379	0.280	0.206	0.349	0.229	0.075
7	0.167	0.296	0.151	0.274	0.064	0.116
8	0.325	0.170	0.310	0.148	0.072	0.105

In particular, when the PSO algorithm was used to perform localization, Table 10 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions. The lower value of  $e_P$  was 0.107 m, and the higher value was 0.503 m, which ensured an accuracy of localization compatible with the intended applications discussed in Section 1. Moreover, the lower value of the standard

deviation was 0.045 m, and the higher value was 0.589 m. Such values of  $e_P$  and  $\sigma_P$  guaranteed accurate position estimates. The values of  $e_P^x$  and  $\sigma_P^x$  were similar to the corresponding values of  $e_P^y$  and  $\sigma_P^y$ , which ensured that the position estimates were spread all around the true position of the TN and that the position estimates were not distributed along special directions.

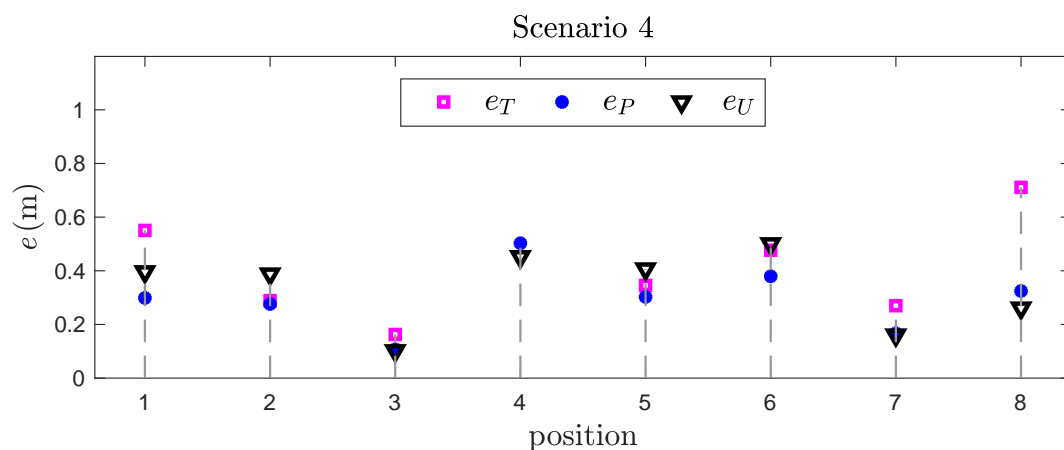
Finally, when the  $POST_U$  algorithm was used to perform localization, Table 11 shows, for each one of the eight positions of the TN in Figure 1, the values of the average localization errors and of the standard deviations of the localization errors computed over the  $r$  acquisitions.

**Table 11.** Measured accuracy of the  $POST_U$  algorithm in Scenario 4.

Position	$e_U$ (m)	$\sigma_U$ (m)	$e_U^x$ (m)	$\sigma_U^x$ (m)	$e_U^y$ (m)	$\sigma_U^y$ (m)
1	0.396	0.124	0.213	0.141	0.313	0.094
2	0.388	0.103	0.173	0.145	0.327	0.053
3	0.102	0.166	0.089	0.158	0.028	0.065
4	0.452	0.463	0.391	0.423	0.155	0.251
5	0.407	0.117	0.250	0.126	0.307	0.081
6	0.500	0.311	0.306	0.401	0.276	0.129
7	0.160	0.287	0.151	0.277	0.039	0.084
8	0.261	0.055	0.220	0.073	0.070	0.111

The lower value of  $e_U$  was 0.102 m, and the higher value was 0.500 m, which ensured a high level of localization accuracy. Table 11 also shows that the lower value of the standard deviation  $\sigma_U$  was 0.055 m, and the higher value was 0.463 m. Such values of  $e_U$  and  $\sigma_U$  guaranteed accurate position estimates that were compatible with the envisaged applications, as discussed in Section 1. As expected, the values of  $e_U^x$  and  $\sigma_U^x$  were similar to the corresponding values of  $e_U^y$  and  $\sigma_U^y$ , which ensured that the position estimates were not distributed along special directions.

Figure 5 shows a summary of the experimental results for the fourth scenario. Note that the values obtained with the three localization algorithms were all similar in this scenario. Therefore, in the idealized case of an empty environment, the three considered algorithms achieved similar levels of accuracy, which were all sufficient to support the envisaged applications of indoor localization.



**Figure 5.** The average errors in Scenario 4 for the TSML algorithm ( $e_T$ ), the PSO algorithm ( $e_P$ ), and the  $POST_U$  algorithm ( $e_U$ ) are shown in meters.

## 5. Conclusions

This paper proposed the  $POST_U$  algorithm in the context of the localization as optimization approach. The  $POST_U$  algorithm was inspired by nonlinear programming, and it could be used to solve the optimization problem obtained from a localization problem with guarantees in terms of the accuracy and of the robustness of localization. In particular, the  $POST_U$  algorithm estimated the position of a TN by means of appropriate subdivisions of the considered environment to search for a

position in the environment that corresponded to the solution of the optimization problem obtained from the considered localization problem. The major assumptions of the  $POST_U$  algorithms were the same as were adopted by many other localization algorithms. The  $POST_U$  algorithm assumed that a sufficient number of ANs were located in the environment at fixed and known positions and that the TN could actively measure estimates of the distances from each AN using an underlying ranging technology. In addition, in its simplest form, the  $POST_U$  algorithm assumed that the considered environment could be split into possibly overlapping boxes, so that the adopted optimization algorithm could be restricted to work on boxes. Observe that the performance of the  $POST_U$  algorithm did not depend on the positions of the ANs, and it only depended on the precision and on the accuracy of the estimates of the distances that the underlying ranging technology provided.

The accuracy of the  $POST_U$  algorithm was studied empirically using a prototype implementation that processed distance estimates from four ANs in an empty corridor. The distance estimates were acquired using UWB signaling from a set of ANs installed specifically to support localization. The obtained experimental results confirmed that the accuracy of the  $POST_U$  algorithm in the studied scenarios was compatible with the most common applications of indoor localization, for which the expected accuracy was normally set to 1 m. The obtained experimental results also showed that the accuracy of the  $POST_U$  algorithm could be superior to the accuracy of the well known TSML algorithm, which is often used as a reference for the assessment of the performance of localization algorithms because it can attain the Cramér–Rao lower bound for the position estimator (e.g., [20]). In particular, the TSML algorithm was not applicable in the first experimental scenario, in which the ANs were all positioned at the same height near the ceiling of the corridor, while the position estimates obtained using the  $POST_U$  algorithm in such a scenario were accurate. Unfortunately, such an arrangement of the ANs is one of the most common in indoor installations because it ensures a good line of sight coverage, which reduces the errors associated with the measurement of distances. In the second experimental scenario, the TSML algorithm was technically applicable, but it was largely outperformed by the  $POST_U$  algorithm. Actually, the experimental results confirmed that the accuracy of the TSML algorithm degraded so significantly when the ANs were installed in critical arrangements that it became practically inapplicable in such cases. In the last two experimental scenarios, in which the TSML algorithm was applicable with sufficient accuracy, the highest average errors of the  $POST_U$  algorithm were lower than the corresponding errors of the TSML algorithm, which made the  $POST_U$  algorithm preferable also in such scenarios. Finally, note that the performance of the PSO algorithm and the performance of the  $POST_U$  algorithm were similar in all considered scenarios, but the  $POST_U$  algorithm was preferable because of two important reasons. First, it guaranteed that the globally optimal solution of the optimization problem was found with a given localization tolerance. Second, it did not depend on the values of configuration parameters, except for the accepted localization tolerance. On the contrary, the PSO algorithm did not ensure that the globally optimal solution of the optimization problem was found, and it used a rich set of configuration parameters whose values needed to be accurately chosen to guarantee good performance.

The experimental results discussed in this paper were obtained using UWB signaling to allow the TN to measure the distances from available ANs. While UWB is appreciated for its accuracy and robustness, a network of UWB nodes acting as ANs is needed to support localization, which is a demanding requirement for the ubiquitous use of indoor localization. This is the reason why preliminary studies on the use of the localization as optimization approach with WiFi signaling, instead of UWB signaling, were performed (e.g., [11,12]). A WiFi infrastructure is already available in the large majority of the indoor environments in which indoor localization could be relevant for the envisaged applications, and therefore, the use of WiFi signaling as the underlying ranging technology could benefit from existing infrastructures, rather than requiring dedicated, and often expensive, infrastructures. Unfortunately, when WiFi signaling is chosen, the problems caused by the arrangement of the ANs in the environment are particularly relevant. The ANs, which are the WiFi access points when WiFi signaling is used, are normally already installed in the environment, and they



are typically located on regular grids near the ceiling of the considered environment, which makes the TSM algorithm practically inapplicable. This is the reason why the  $POST_U$  algorithm, and possibly other algorithms based on the localization as optimization approach, can play a crucial role for the adoption of WiFi signaling to support ubiquitous indoor localization.

Future developments of the discussed research concern empirical studies of the characteristics of the proposed algorithm in environments that contain obstacles and people in order to better assess the performance of the algorithm in real applications. A second planned development of the discussed research is intended to use the information on the movements of the TN, which is typically available when smartphones are used, to predict the positions of the TN, and to use predicted positions to improve the accuracy and the robustness of localization algorithms. A third development concerns the possibility of studying and experimenting with hybrid approaches to localization designed to integrate the approach based on active ranging adopted in this work with approaches based on active or passive tags. Finally, the algorithm proposed in this paper is inspired by a specific technique for nonlinear optimization, but other valid alternatives documented in the vast literature on the subject can be examined, especially when the considered environments have intricate structures.

**Author Contributions:** These authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Zafari, F.; Gkelias, A.; Leung, K.K. A survey of indoor localization systems and technologies. *IEEE Commun. Surv. Tutorials* **2019**, *21*, 2568–2599. [[CrossRef](#)]
2. Bergenti, F.; Caire, G.; Gotta, D. An overview of the AMUSE social gaming platform. In Proceedings of the 14th Workshop “From Objects to Agents” (WOA 2013), Torino, Italy, 2–3 December 2013; Volume 1099, pp. 85–90.
3. Monica, S.; Bergenti, F. Location-aware social gaming with AMUSE. In Advances in Practical Applications of Scalable Multi-Agent Systems; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9662, pp. 36–47.
4. Monica, S.; Bergenti, F. Experimental evaluation of agent based localization of smart appliances. In Proceedings of the 14th European Conference on Multi-Agent Systems (EUMAS 2016), Valencia, Spain, 15–16 December 2016; Springer: Berlin/Heidelberg, Germany, 2017; Volume 10207, pp. 293–304.
5. Purushotham, S.; Kuo, C.C.J. Personalized group recommender systems for location- and event based social networks. *ACM Trans. Spat. Algorithms Syst.* **2016**, *2*, 16:1–16:29. [[CrossRef](#)]
6. Skoumas, G.; Pfoser, D.; Kyrillidis, A.; Sellis, T. Location estimation using crowdsourced spatial relations. *ACM Trans. Spat. Algorithms Syst.* **2016**, *2*, 5:1–5:23. [[CrossRef](#)]
7. Teng, S.Y.; Ku, W.S.; Chuang, K.T. Toward mining stop-by behaviors in indoor space. *ACM Trans. Spat. Algorithms Syst.* **2017**, *3*, 7:1–7:38. [[CrossRef](#)]
8. Mendoza-Silva, G.M.; Torres-Sospedra, J.; Huerta, J. A meta-review of indoor positioning systems. *Sensors* **2019**, *19*, 4507. [[CrossRef](#)] [[PubMed](#)]
9. Pannuto, P.; Kempke, B.; Chuo, L.X.; Blaauw, D.; Dutta, P. Harmonium: Ultra wideband pulse generation with bandstitched recovery for fast, accurate, and robust indoor localization. *ACM Trans. Sens. Netw.* **2018**, *14*, 11:1–11:29. [[CrossRef](#)]
10. Shi, G.; Ming, Y. Survey of indoor positioning systems based on ultra-wideband (UWB) technology. In *Wireless Communications, Networking and Applications*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 348, pp. 1269–1278.
11. Monica, S.; Bergenti, F. Indoor localization of JADE agents without a dedicated infrastructure. In *Multiagent System Technologies*; Springer: Berlin/Heidelberg, Germany, 2017; Volume 10413, pp. 256–271.
12. Monica, S.; Bergenti, F. Optimization based robust localization of JADE agents in indoor environments. In Proceedings of the 3rd Italian Workshop on Artificial Intelligence for Ambient Assisted Living (AI\*AAL.it 2017), Bari, Italy, 16–17 November 2017; Volume 2061, pp. 58–73.

13. Mekelleche, F.; Haffaf, H. Classification and comparison of range based localization techniques in wireless sensor networks. *J. Commun.* **2017**, *12*, 221–227. [[CrossRef](#)]
14. Ho, K.C.; Xiaoning, L.; Kovavisaruch, L. Source localization using TDOA and FDOA measurements in the presence of receiver location errors: Analysis and solution. *IEEE Trans. Signal Process.* **2007**, *55*, 684–696. [[CrossRef](#)]
15. Monica, S.; Ferrari, G. A swarm based approach to real-time 3D indoor localization: Experimental performance analysis. *Appl. Soft Comput.* **2016**, *43*, 489–497. [[CrossRef](#)]
16. Monica, S.; Ferrari, G. Impact of the number of beacons in PSO based auto-localization in UWB networks. *Applications of Evolutionary Computation*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 7835, pp. 42–51.
17. Monica, S.; Ferrari, G. Swarm intelligent approaches to auto-localization of nodes in static UWB networks. *Appl. Soft Comput.* **2014**, *25*, 426–434. [[CrossRef](#)]
18. Bonyadi, M.R.; Michalewicz, Z. Particle swarm optimization for single objective continuous space problems: A review. *Evol. Comput.* **2017**, *25*, 1–54. [[CrossRef](#)] [[PubMed](#)]
19. Bergenti, F.; Monica, S. Satisfaction of polynomial constraints over finite domains using function values. In Proceedings of the 18th Italian Conference on Theoretical Computer Science (ICTCS 2017) and the 32nd Italian Conference on Computational Logic (CILC 2017), Naples, Italy, 26–28 September 2017; Volume 1949, pp. 262–275.
20. Álvarez, R.; Díez-González, J.; Alonso, E.; Fernández-Robles, L.; Castejón-Limas, M.; Perez, H. Accuracy analysis in sensor networks for asynchronous positioning methods. *Sensors* **2019**, *19*, 3024. [[CrossRef](#)] [[PubMed](#)]
21. Bellifemine, F.; Bergenti, F.; Caire, G.; Poggi, A. JADE—A Java agent development framework. In *Multi-Agent Programming*; Springer: Berlin/Heidelberg, Germany, 2005; Volume 15, pp. 125–147.
22. Bergenti, F. An introduction to the JADEL programming language. In Proceedings of the 26th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2014), Limassol, Cyprus, 10–12 November 2014; pp. 974–978.
23. Monica, S.; Bergenti, F. Location-aware JADE agents in indoor scenarios. In Proceedings of the 16th Workshop “From Objects to Agents” (WOA 2015), Naples, Italy, 17–19 June 2015; Volume 1382, pp. 103–108.
24. Bergenti, F.; Ricci, A. Three approaches to the coordination of multiagent systems. In Proceedings of the 17th ACM Symposium on Applied Computing (SAC 2002), Madrid, Spain, 11–14 March 2002; pp. 367–372.
25. Monica, S.; Ferrari, G. 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 2013), Lausanne, Switzerland, 4–6 April 2013; Volume 7824, pp. 456–465.
26. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell. J.* **2007**, *1*, 33–57. [[CrossRef](#)]
27. Bergenti, F.; Monica, S. Experiments on robust indoor localization of mobile devices using interval arithmetic. In Proceedings of the 20th Workshop “From Objects to Agents” (WOA 2019), Parma, Italy, 26–28 June 2019; Volume 2402, pp. 14–21.
28. Farouki, R.T. The Bernstein polynomial basis: A centennial retrospective. *Comput. Aided Geom. Des.* **2012**, *29*, 379–419. [[CrossRef](#)]
29. Titi, J.; Garloff, J. Matrix methods for the tensorial Bernstein form. *Appl. Math. Comput.* **2019**, *346*, 254–271. [[CrossRef](#)]
30. Bergenti, F.; Monica, S. Hyper-arc consistency of polynomial constraints over finite domains using the modified Bernstein form. *Ann. Math. Artif. Intell.* **2017**, *80*, 131–151. [[CrossRef](#)]

