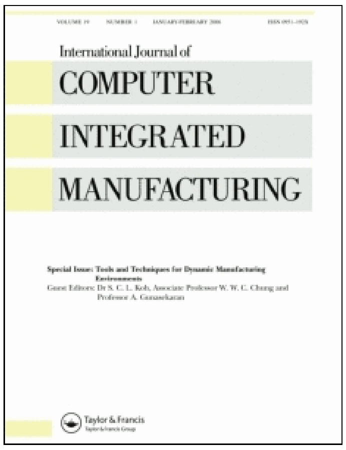


International Journal of Computer Integrated Manufacturing
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713804665

A review of localization algorithms for distributed wireless sensor networks in manufacturing

F. Franceschini [a]; M. Galetto [a]; D. Maisano [a]; L. Mastrogiacomo [a]
[a] Dipartimento di Sistemi di Produzione ed Economia dell'Azienda, Politecnico di Torino, Italy

First Published:July2009

To cite this Article Franceschini, F., Galetto, M., Maisano, D. and Mastrogiacomo, L.(2009)'A review of localization algorithms for distributed wireless sensor networks in manufacturing',International Journal of Computer Integrated Manufacturing,22:7,698 — 716
To link to this Article: DOI: 10.1080/09511920601182217
URL: http://dx.doi.org/10.1080/09511920601182217

International Journal of Computer Integrated Manufacturing
Vol. 22, No. 7, July 2009, 698–716

A review of localization algorithms for distributed wireless sensor networks in manufacturing

F. Franceschini*, M. Galetto, D. Maisano and L. Mastrogiacomo

Dipartimento di Sistemi di Produzione ed Economia dell'Azienda, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy

Wireless sensor networks (WSNs) typically consist of a large number of densely populated sensor nodes. Due to important advances in integrated circuits and radio technologies, the use of distributed sensor networks is becoming increasingly widespread for a variety of applications, e.g. indoor navigation, environmental monitoring, people and object tracking, logistics, industrial diagnostics, quality control, and other manufacturing activities. In many cases, such as in objects tracking, knowing the physical location of network nodes is essential. Locating elements of WSNs is not a trivial task. Manual methods are wearisome and may be inaccurate, especially for large-scale networks. Therefore, many self-locating methods – where nodes cooperate with each other without human involvement – have recently been studied and implemented. The purpose of this work is to analyse the most significant methods for automatic location of distributed WSNs. The first part of the paper provides a description of the most common criteria used to categorize existing network localization algorithms. A taxonomy is then suggested that may be a useful tool to help evaluate, compare and select such algorithms. Five of the most representative algorithms are explained and discussed in detail in order to identify their strong points and their limitations.

Keywords: distributed wireless sensor networks; localization algorithms; wireless networks; algorithm taxonomy; manufacturing

1. Introduction

A wireless network typically consists of a large number of nodes (e.g. sensor devices) with a dense distribution, equipped with transceivers. Each device can communicate with other devices within its communication range. A wireless network is typically modelled as a graph, where each node represents a physical device. Two nodes are connected by an edge, if and only if, they can directly communicate.

Dramatic advances in integrated circuits and radio technologies have made the use of large wireless sensor networks (WSNs) possible for many applications. In particular, attention towards the utilization of WSNs in manufacturing is increasing. Since sensor devices do not need cables and may be easily deployed or moved, they can be practically utilized for a variety of industrial applications – factory logistics and warehousing, environmental control and monitoring, support for assembly processes, industrial dimensional measuring and real-time surveillance are only some possible applications of WSNs (Doss and Chandra 2005, Intel Corporation 2005, Koumpis *et al*. 2005, Pepperl+Fuchs 2005, Franceschini *et al*. 2006, Oh *et al*. 2006, Pan *et al*. 2006, Wang and Xi 2006). While outdoor localization applications are widespread today (e.g. Global Positioning System (GPS)), indoor applications can also benefit from location determination knowledge (Gotsman and Koren 2004). To make such applications feasible, the device costs should be low and the network should be organized without significant human involvement.

The solution of adding a GPS device to all the nodes in a network is not practical for many reasons. GPS devices cannot work indoors, they are bulky, expensive and are inefficient in power consumption, while wireless sensor nodes are required to be small, low priced and low powered (Bulusu *et al*. 2000).

In some applications (e.g. indoor navigation, objects tracking, remote diagnostics, etc.) mobile nodes calculate their position making reference to fixed network nodes. So, fixed network nodes should be aware of their respective location. To reach this state – especially for large-scale sensor networks – many self-localization methods have been recently studied and implemented. Generally, nodes automatically

*Corresponding author. Email: fiorenzo.franceschini@polito.it

ISSN 0951-192X print/ISSN 1362-3052 online
© 2009 Taylor & Francis
DOI: 10.1080/09511920601182217
http://www.informaworld.com

cooperate, estimating local distances to their neighbours, converging to a consistent coordinate assignment. Nodes work together in a peer-to-peer way to build a map of the sensor network.

Received-Signal-Strength (RSS) and Time-of-Arrival (ToA) are two common approaches for estimating the distance between nodes within their mutual transmission range (Wu *et al.* 2005). RSS measures the power of the signal at the receiver and calculates the distance according to the propagation loss model (see Figure 1). ToA measures the propagation time (Δt) of the received signal (typically radio signals for large distances or ultrasound for small distances) and determines the distance by multiplying it by its own speed. In general, RSS is an easier parameter to implement, while ToA may achieve a higher accuracy (Patwari *et al.* 2005).

In current work at the technical laboratory at DISPEA (Politecnico di Torino, Italy) a metrological application based on a WSN has been developed. Such an application requires a reasonable level of accuracy in distance estimates. As a consequence, inter-node distances are measured implementing a ToA technique, with ultrasound transceivers. Considering a speed of sound of around 340 m/s (when temperature and relative humidity of the air are $T = 20°\text{C}$ and $H \approx 50\%$, respectively), a propagation time $\Delta t = 10$ ms corresponds to a distance $D = v\Delta t \approx 3.4$ mm between ultrasound transceivers. (Note that in the same propagation time Δt a radio signal (speed around 300 000 km/s) covers a distance of 3000 km.) The limited resolution of timers is the main reason why ultrasound signals are preferable to radio signals for small distance measurements.

Angle-of-Arrival (AoA) is another approach for WSNs localization. Usually, sensor nodes receive signals from at least three neighbours (in particular, to collect angle information) and determine their coordinates by triangulation according to the angle bearings of incoming signals (Nasipuri and Li 2002, Niculescu and Nath 2003). One potential problem of this approach is the expense of equipment to obtain precise angle estimates (Priyantha *et al.* 2003). Due to the drawbacks of implementing AoA techniques, in the following discussion we assume RSS or ToA approaches to estimate distances between neighbouring nodes.

1.1. *Applications of WSNs in manufacturing*

To give a concrete idea of the potential of WSNs in manufacturing, this section briefly introduces some of the most interesting research issues.

(1) *Support for final assembly*. Ultrasonic sensors are mounted on power tools (e.g. screwdrivers) to detect their real position and activate them if they are in the right position, during final assembly (Pepperl+Fuchs 2005).
(2) *Industrial control and monitoring*. Sensor devices can be deployed to perform industrial control and monitoring (for instance control of the air conditions of pollution, temperature and pressure in different areas of a factory) or for emergency responses in the case of incidents (Doss and Chandra 2005, Koumpis *et al.* 2005, Pan *et al.* 2006).
(3) *Dimensional measuring*. Coordinate measurement of large objects by means of a wireless sensor 'constellation' distributed around them. This research project was developed at the industrial metrology and quality laboratory of DISPEA, Politecnico di Torino (Franceschini *et al.* 2006).
(4) *Factory logistics and warehousing*. In an industrial warehouse mobile forklifts generally move along corridors in order to reach the shelves where goods are stored (see Figure 2). Forklifts and shelves can be equipped with ultrasound transceivers that communicate with each other, with the purpose of evaluating mutual distances using a ToA technique

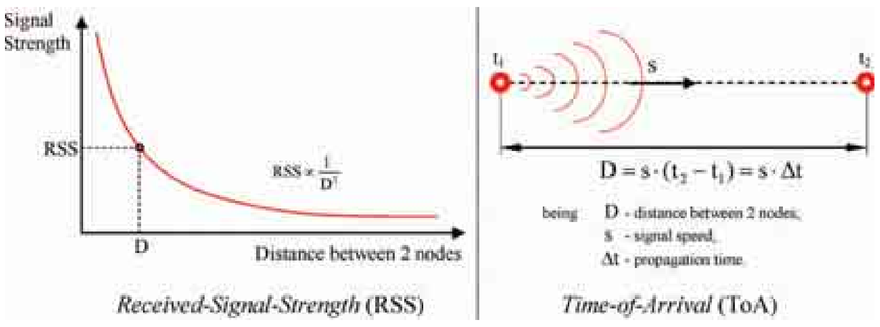

Signal Strength
RSS
$RSS \propto \frac{1}{D^2}$
D
Distance between 2 nodes
Received-Signal-Strength (RSS)
t_1
s
t_2
$D = s \cdot (t_2 - t_1) = s \cdot \Delta t$
being D - distance between 2 nodes;
s - signal speed,
Δt - propagation time
Time-of-Arrival (ToA)

Figure 1. Representation scheme of RSS and ToA approaches for distance estimation.

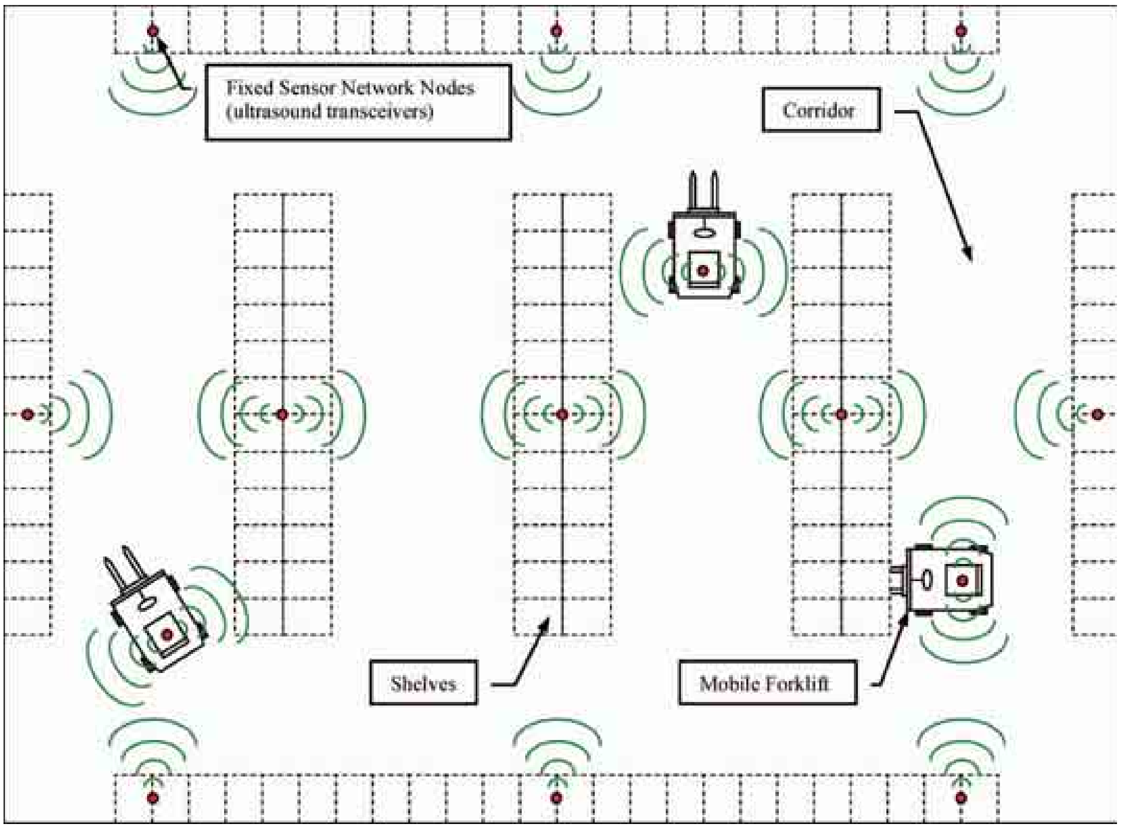


Figure 2. Schematic layout: industrial warehouse equipped with a network of wireless sensors.

(Intel Corporation 2005). This type of WSN can be utilized to calculate the position of the forklifts for:

- indoor navigation – mobile forklifts, equipped with wireless transceiver, can be automatically guided towards their destination;
- traffic monitoring – physical traffic can be monitored in order to identify most congested areas or to improve goods distribution (Capkun *et al.* 2001).

2. Scope and method of the review

The purpose of this paper is to provide a reference framework of the major algorithms for automatic localization of network nodes. A taxonomy to evaluate and compare them is also suggested. The first part of the paper provides a description of the most common criteria to categorize network localization algorithms. Subsequently, five of the most representative algorithms are independently described and set in the suggested taxonomy, in order to identify their common features as well as those that set them apart. Considering the great abundance of algorithms presented in the literature, those discussed here were selected owing to their originality and spread. Finally, other network localization algorithms are briefly described. All algorithms are accompanied by explanatory representation schemes.

2.1. Categorization of network localization algorithms

Generally, localization algorithms are designed to be applied to a typical sensor network comprising a large number of nodes with a dense distribution. As a consequence, many of them do not fit to small networks with few distributed nodes. In this latter case, nodes can be manually located. Localization algorithms can be classified into the following four categories.

(1) The first category is based on the presence (or absence) of nodes with pre-configured coordinates.
   - *Anchor-based algorithms*. The location system is implemented by selecting a set of reference nodes ('landmarks' or 'anchor nodes') with known coordinates. A localization system with anchor nodes has the

limitation that it needs another location system (e.g. GPS) to determine the anchor nodes' positions. Furthermore, a large number of anchor nodes is required for the resulting position errors to be acceptable (Priyantha *et al.* 2003).
- *Anchor-free algorithms*. These use local distance measurements among nodes to determine their respective coordinates; they do not assume the availability of nodes with pre-configured coordinates.

(2) The second category is based on the way node locations 'propagate' in the network.
- *Incremental algorithms*. These algorithms usually start with a set of three or more reference nodes with known coordinates. Other nodes in the network can contact the reference nodes and determine their own coordinates. As an unknown position node obtains an acceptable position estimate, it may serve as a new reference point. This process can be incrementally applied until all nodes in the network have obtained their coordinates.
- *Concurrent algorithms*. In this approach, many pairs of sensors communicate and share measurements in order to achieve localization for all sensors. Rather than solving each sensor position one at time, all sensor positions are simultaneously estimated. Such localization systems not only allow unknown-location devices to make measurements with known-location references, but they additionally allow unknown-location devices to make measurements with other unknown-location devices. The additional information gained from these measurements between pairs of unknown-location devices enhances the accuracy and robustness of the localization system. Such systems have been described as 'co-operative' (Ji and Zha 2004, Patwari *et al.* 2005).

(3) The third category subdivides localization approaches into two broad classes, based on the 'granularity' of information acquired by the sensors during communication.
- *Fine-grained algorithms*. Algorithms that use accurate information – such as the *distance* from a reference point based on RSS or ToA measurements – fall into the category of fine-grained localization methods. Typically, they use technologies, such as infrared (IR), ultrasound (US) or radio frequency (RF) signals.
- *Coarse-grained algorithms*. Algorithms that utilize less accurate information, such as *proximity* (two devices are considered to be 'in proximity' if they can directly communicate) to a given reference point, are categorized as coarse-grained localization methods. Coarse-grained algorithms estimate inter-node distances using rough techniques such as hop-count. In a wireless network, the number of hops is the number of edges traversed by a signal along the shortest path between the source node and the destination node. For example, in Figure 3 the number of hops between nodes *j* and *n* is 2. Hop-count may be used to determine a rough evaluation of inter-node distances (Priyantha *et al.* 2003).

As expected, fine-grained algorithms are more accurate than coarse-grained. In the absence of measurement errors, fine-grained algorithms provide exact network nodes positioning.

(4) The fourth category is based on computational distribution.
- *Centralized algorithms*. Computing is performed by a single centralized node or network device. All nodes broadcast information to a single computer to solve the localization problem (Doherty *et al.* 2001).
- *Distributed algorithms*. Computing is equally distributed among network nodes. Each node receives location information from neighbouring nodes, performs computation and re-transmits the obtained results to them.

It is important to note that many of the algorithms discussed in the following sections have never been physically implemented on real sensor networks. Rather, most of them have been studied and developed on the basis of computer simulations. Few algorithms have been practically tested in WSNs. The difficulty

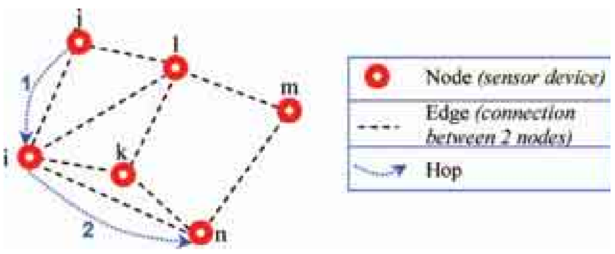


Figure 3. Schematic representation of the concept of hop-count in a sensor network.

with such experimental campaigns is due to the following main aspects: sensor firmware programming, sensor physical allocation, time taken to adjust the network and time for experiments (Patwari *et al*. 2005). Regarding the future, additional effort is needed to test algorithms in order to practically assess their performance and reliability.

2.2. *Taxonomy description*

In this section we propose a taxonomy to benchmark network localization algorithms. Taxonomy is a useful tool for evaluating and comparing algorithms, depending on the network features and peculiarities. In the next section, five of the most representative localization techniques are illustrated and classified in detail. Evaluation criteria are defined and described in Table 1.

3. Detailed description of localization algorithms

In this section five of the most significant *fine-grained* localization algorithms are described in detail, following the criteria on taxonomy presented previously. The descriptions of the algorithms are summarized in Table 2 to assist network designers in their evaluation and comparison.

3.1. *Assumption based coordinates (ABC) algorithm*

The ABC algorithm is a 2D/3D, *incremental* and *anchor-free* algorithm (Savarese *et al*. 2001). It starts with a node (n_0) assuming that it is located at the origin of a local coordinate system. The algorithm localizes three (two in 2D networks) other nodes directly connected with n_0, assigning them coordinates in order to satisfy the inter-node distances (these nodes (n_1,n_2, n_3) are the first to establish a connection with n_0.) To build such a local coordinate system the following assumptions are made:

(1) n_1 is located along the x-axis;
(2) the direction of the positive y-axis is defined by n_2;
(3) the direction of the positive z-axis is defined by n_3 (see Figure 4).

The algorithm proceeds incrementally. Given a new node with unknown position, it calculates its coordinates using the distances to four (or more) neighbours

Table 1. Definitions and descriptions of the suggested taxonomy.

	Criterion	Description
Algorithm description	Name/acronym	Name or acronym assigned by the author(s)
	Author(s) and publication date	Author(s) name(s) and date of the algorithm official release
	Fine-grained/coarse-grained	'Granularity' of the inter-node distance estimates provided by the algorithm
	Short description	Short description of the algorithm *modus operandi*
Network features	2D/3D	Space displacement of the sensors networks (2D if nodes are coplanar, 3D if they are spatially distributed)
	Single-hop/ multi-hop	In single-hop networks all sensors are connected to each other. In multi-hop networks, not all the sensors are directly connected. They can communicate using specific routing protocols
	Limitations	Specific restrictions or features of the network (e.g. node distribution)
	Anchor-free/anchor-based	Anchor-free algorithms do not require nodes with pre-configured coordinates
Computational workload	Data processing description	Short description of data processing method.
	Centralized/distributed algorithm	Computing is performed by a single centralized node or network device, or it is equally distributed among network nodes
	Incremental/concurrent algorithm	Nodes positions are incrementally (one after the other), or concurrently (parallel processing) estimated
	Computational complexity	Quantitative evaluation of the time required during computation. Generally, it is estimated depending on number of nodes, network connectivity[†], or other network parameters
Benefits		Best advantages in using the algorithm
Drawbacks		Major deficiencies and drawbacks of the algorithm
Possible improvements		Possible ways of addressing the problems and limitations of the algorithm

[†]From network theory, *connectivity* between two nodes is defined as the number of connections (edges) in the network allowed to fail before the two nodes (vertices) become disconnected. Network connectivity is defined as the mean value of the network connectivities.

Table 2. Taxonomy of localization algorithms.

	Name/Acronym	ABC algorithm	TERRAIN algorithm	Savvides *et al.* algorithm	AFL algorithm	Moore *et al.* algorithm
Algorithm Description	Author(s) and public. date	Savarese *et al.* (2001)	Savarese *et al.* (2001)	Savvides *et al.* (2001)	Priyantha *et al.* (2003)	Moore *et al.* (2004)
	Fine-grained/ coarse-grained	Fine-grained	Fine-grained	Fine-grained	First phase is coarse-grained but, globally, the algorithm is fine-grained	Fine-grained
	Short description	ABC first selects four in-range nodes and assigns them coordinates to satisfy the inter-node distances. Then the coordinates of other nodes are incrementally calculated using the distances from (at least) four nodes with already calculated coordinates	Algorithm which builds on ABC. Each anchor starts the ABC algorithm. Using the coordinates assigned by ABC, each node can calculate the distances to at least four localized nodes. Then it localizes itself in a general coordinate system, performing a triangulation	A node is localized depending on the positions of the already localized neighbours (at least four). Position is estimated performing a Maximum Likelihood optimization. The process can be applied recursively until all nodes in the network have obtained their coordinates	AFL is based on two phases. First phase coarsely estimate the network's global layout by using a hop-count technique. Second phase optimizes this layout implementing a mass-spring relaxation, based on more accurate inter-node distances, measured using ToA	In the first phase each node becomes the centre of a *cluster* and estimates the relative location of the neighbours which can be unambiguously localized. Second phase is an optional optimization. Third phase is implemented using a 'cluster stitching' technique to obtain a coordinate assignment for all the nodes, within a general coordinates system
Network Features	2D/3D	2D,3D	2D,3D	2D,3D	2D,3D	2D only. The algorithm has not been implemen-ted yet in 3D case
	Single-hop/ multi-hop	Single-hop and multi-hop	Single-hop and multi-hop	Single-hop and multi-hop	Multi-hop only. In one-hop networks – where all nodes are connected each other – it fails because the first phase, based on the hop-count, can not be executed	Single-hop and multi-hop
	Limitations	To calculate its coordinates, each node with unknown position should communicate with (at least) four non-coplanar nodes – in 3D – or three non-aligned nodes – in 2D – with already calculated coordinates	–	To calculate its coordinates, each node with unknown position should communicate with (at least) four non-coplanar nodes – in 3D – or three non-aligned nodes – in 2D – with already calculated coordinates	–	–
	Anchor-free/ anchor-based	Anchor-free	Anchor-based	Anchor-based	Anchor-free	Anchor-free

(*continued*)

Table 2. (*Continued*).

	Name/Acronym	ABC algorithm	TERRAIN algorithm	Savvides *et al.* algorithm	AFL algorithm	Moore *et al.* algorithm
Computational Workload	Data processing description	Every node repeatedly (1) receives location information from neighbouring nodes; (2) solves a local optimization problem; (3) transmits the obtained results to the neighbouring nodes	This algorithm, like ABC, is based on the solution of triangulation problems	Each node runs an optimization, based on Maximum Likelihood, as soon as it receives distance measurements from already localized neighbouring nodes (at least four). Once localized it broadcasts its position to neighbours	First phase is based on hop-count. Second phase is a mass-spring optimization. Every node repeatedly solves a local optimization problem and transmits the obtained results to neighbouring nodes. This optimization requires multiple iterations	First phase of the algorithm is based on triangulation, preceded by a non-ambiguity testing. Second phase is a mass-spring relaxation, analogous to the AFL one, to refine the localization of the clusters. In third phase, clusters are stitched using a closed form solution for a least-squares problem
	Centralized/ distributed algorithm	Distributed algorithm. Centralized computation is not required	Distributed algorithm. Centralized computation is not required	The algorithm can work both in a distributed or centralized way	First phase of AFL is centralized, because it can hardly be implemented without a centralized device handling information from nodes. Second phase is a distributed optimization	'Cluster stitching' can hardly be implemented without a centralized network device, which handles information from clusters that should be stitched together
	Incremental/ concurrent	Incremental	Incremental	Incremental	Concurrent	Concurrent
	Computational complexity	Each node performs O(n) computations, with n the number of already localized neighbours	Computational complexity of each node is higher than ABC. We estimated it to be in the order of O[$m(n+1)$], where n is the number of neighbours and m the number of ABC algorithms that have propagated to the node	Each node performs O(n) computations, being n the number of already localized neighbours	AFL computational complexity tightly depends on the computational complexity of mass-spring optimization. For the single node, we estimated it to be in the order of $O(m \cdot n)$, where n is the number of neighbours and m the number of required iterations	Authors show that for each node the computation is proportional to the third power of the number of neighbours, so it is $O(n^3)$
Benefits		The algorithm is relatively simple and does not require complicate calculations. No anchor-nodes are required	Compared to ABC, this method reduces the error propagation. Error is balanced by a refinement process after nodes localization	The advantage of this method is that the algorithm, in the non-centralized version, is fully distributed and does not require complicate calculations	AFL is anchor-free and does not require nodes with pre-configured coordinates. Instead of incremental algorithms, AFL performs much better, even for networks with small connectivity. Furthermore, AFL error propagation is small	Error propagation is reduced compared to approaches based on pure triangulation. Simulations show that error on node positioning, using incremental methods, is significantly higher

(*continued*)

Table 2. (*Continued*).

Name/Acronym	ABC algorithm	TERRAIN algorithm	Savvides *et al.* algorithm	AFL algorithm	Moore *et al.* algorithm
Drawbacks	ABC suffers from error propagation. Positioning accuracy decreases moving away from the node which started the algorithm. The algorithm can be inaccurate, especially for widespread networks. Because of its incremental nature, the complete graph realization is not always guaranteed. If measurements are corrupted by noise, the algorithm can lead to incorrect nodes localizations	Although TERRAIN is more accurate than ABC, it still suffers from error propagation. Complete graph realization is not always guaranteed. If measurements are corrupted by noise, the algorithm can lead to incorrect nodes localizations	The algorithm suffers from error accumulation. In the centralized version, error propagation is slightly reduced. Complete graph realization is not always guaranteed. If measurements are corrupted by noise, the algorithm can lead to incorrect nodes localizations	Simulations have demonstrated that a pure mass-spring algorithm does not work well, producing a network with an incorrect layout, without good initial position estimates. Even if AFL outperforms incremental algorithms, there is not a proof of correctness. AFL may falsely converge to distorted configurations of the network nodes	Under conditions of low node connectivity or high measurement noise, the algorithm may be unable to localize a useful number of nodes. In 3D, computational complexity and data routing dramatically increase
Possible improvements	A partial solution to error propagation consists in introducing a number of anchor-nodes. That will also solve the problem of network orientation, and reduce the risk of incorrect nodes displa-cements	A first solution to prevent error propagation is to increase the number of anchor-nodes. The price to pay is the a-priori localization of such nodes. A different kind of approach leads to the introduction of an iterative refinement process (Savarese *et al.* 2001)	Error propagation can be minimized through an iterative refinement process – for example, a numerical optimization such as mass-spring relaxation (see AFL algorithm) – performed after the nodes location	Present and future improvements are focused on enhancing the first phase. In the actual version, the algorithm lacks a method to prevent realization ambiguities. Research effort focuses on a possible way to realize a completely distributed first phase (Gotsman and Koren 2004)	When robust quadrilaterals do not exist or when the connectivity is poor, Moore *et al.* algorithm fails. To get over these difficulties, the algorithm can be enhanced implementing a more effective robustness test, such as the one proposed by Sottile and Spirito (2006)

with already known coordinates. In general, the trilateration problem can be formulated as follows. Given a set of nodes n_i with known coordinates (x_i, y_i, z_i) and a set of measured distances D_i, a system of equations needs to be solved to calculate the unknown position of $P(u, v, w)$ (Ward *et al.* 1997, Chen *et al.* 2003).

$$\begin{bmatrix} (x_1-u)^2 & (y_1-v)^2 & (z_1-w)^2 \\ (x_2-u)^2 & (y_2-v)^2 & (z_2-w)^2 \\ & \vdots & \\ (x_n-u)^2 & (y_n-v)^2 & (z_n-w)^2 \end{bmatrix} = \begin{bmatrix} D_1^2 \\ D_2^2 \\ \vdots \\ D_n^2 \end{bmatrix} \tag{1}$$

If the trilateration problem is over defined (more equations than required to solve the localization problem), it can be solved using a least-mean squares approach (Savvides *et al.* 2001). The accuracy strongly depends on the geometry of the position references and the accuracy of distance measurements. Errors can propagate through all subsequent trilateration computations, leading to an inaccurate localization of nodes far away from n_0.

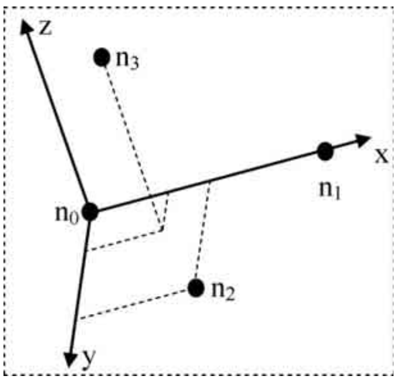


Figure 4. Local coordinate system built around the starting node (n_0).

3.1.1. *Network features*

ABC is an *anchor-free* algorithm developed both for 2D and 3D network topologies. For widespread networks it can be inaccurate due to error propagation. To be located, in a 3D case each node has to communicate with at least four non-coplanar nodes with already known coordinates (three non-aligned in the 2D case, as shown in Figure 5).

3.1.2. *Computational workload*

The position estimation does not require centralized computation. All nodes are not required to communicate their connectivity information to a centralized computer in order to solve the localization problem. Computing is distributed among nodes with each:

(1) receiving ranging and location information from neighbouring nodes;
(2) solving a local localization problem;
(3) transmitting the results to neighbouring nodes.

The computational complexity for each node linearly increases with the number of localized neighbours. Each node performs $O(n)$ computations, n being the number of neighbours already located.

3.1.3. *Benefits*

The algorithm is relatively simple and does not require complicated calculations. Furthermore, no anchor nodes are required.

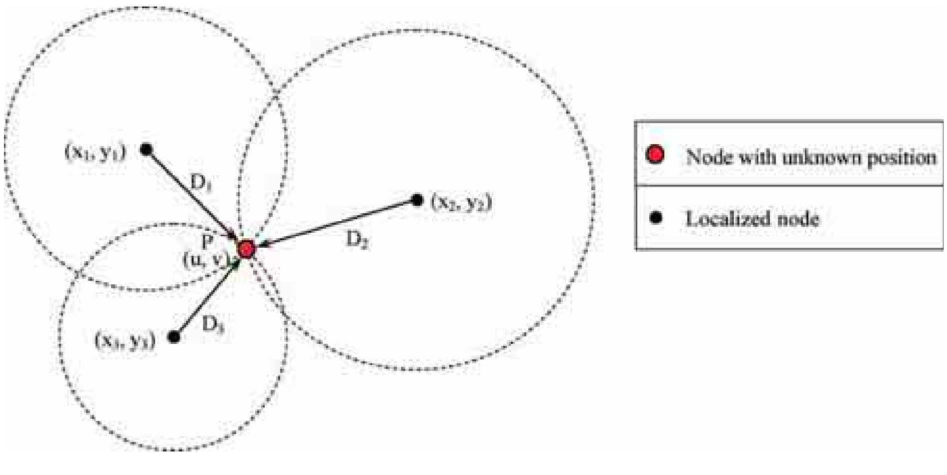


Figure 5. Representation scheme of the trilateration problem in a 2D network.

3.1.4. *Drawbacks*

ABC suffers from error propagation, the results being unsuitable for widespread networks. As with all incremental algorithms, error propagation is cumulative, which results in poor coordinate assignment. In particular, positioning accuracy decreases for nodes that are distant from the 'origin' node. Because of its incremental nature, complete graph realization is not guaranteed even if every node of the network has four neighbours (Priyantha 2003).

If measurements are corrupted by noise, however small, the algorithm can lead to ambiguous or incorrect nodes displacements. Figure 10 (see section 3.5) shows an example of a possible ambiguity.

As with all anchor-free algorithms, ABC will produce a topologically correct map with a random orientation relative to a global coordinate system. In fact, there are an infinite number of network solutions, since the coordinates can be rotated or translated as long as their distances do not change (Gotsman and Koren 2004).

3.1.5. *Possible improvements*

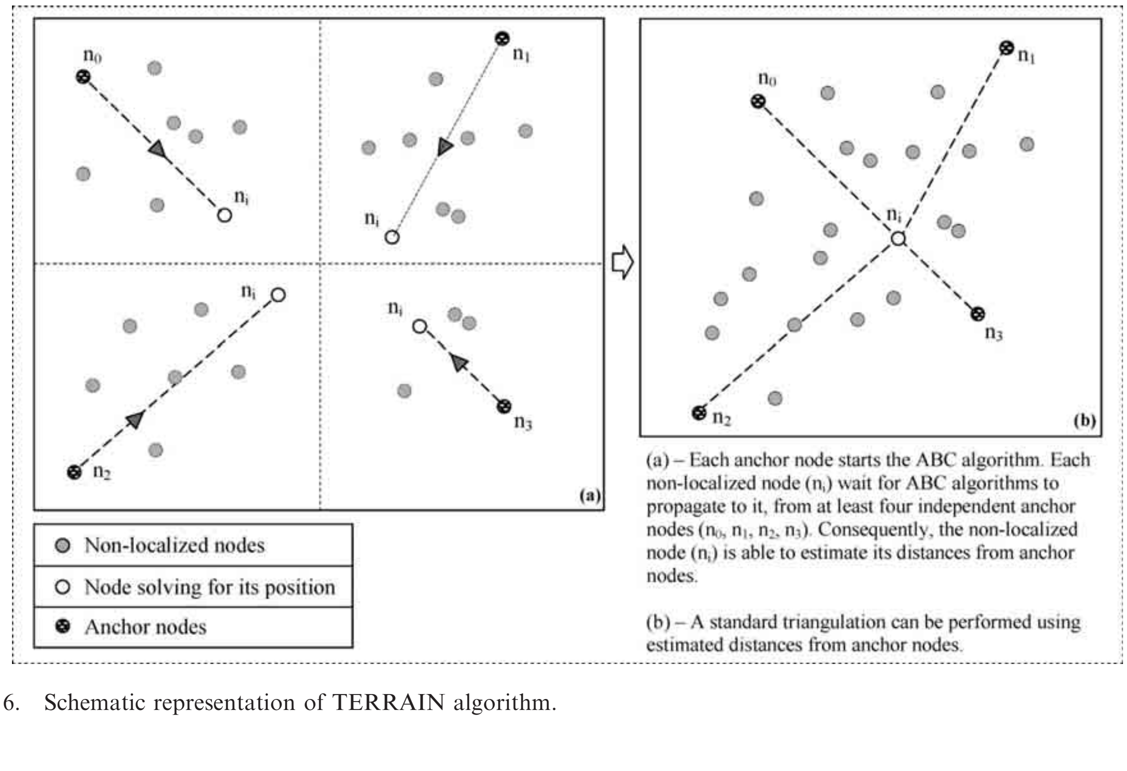
A partial solution to error propagation consists of introducing a number of anchor nodes. Since a global coordinate system is implicitly defined assigning anchor node positions, the problem of network orientation is solved. In addition, the risk of incorrect nodes displacements is reduced. The price to pay for the introduction of anchor nodes is the *a priori* manual location of them.

3.2. *Triangulation via extended range and redundant association of intermediate nodes (TERRAIN) algorithm*

The TERRAIN algorithm builds on the ABC algorithm, but it is anchor-based (Savarese *et al.* 2001, Savarese *et al.* 2002). Nodes are divided into two categories:

(1) *anchor nodes*: reference nodes with known coordinates; to start the algorithm, there must be at least four.
(2) *regular nodes*: other nodes, originally with unknown coordinates.

At first, each anchor node starts executing an independent ABC algorithm (see Figure 6). As a consequence, the number of different ABC algorithms, which will propagate within the network, corresponds to the number of anchor nodes. Furthermore, each ABC assumes that the starting anchor node is located at the origin of a local coordinate system. As explained in section 3.1, such a coordinate system is defined by selecting and localizing the next three regular nodes. Then the algorithm incrementally proceeds. Regular nodes calculate their coordinates, according to the



n_0
n_i
n_1
n_2
n_3
(a)
(b)
● Non-localized nodes
○ Node solving for its position
⊗ Anchor nodes
(a) – Each anchor node starts the ABC algorithm. Each non-localized node (n_i) wait for ABC algorithms to propagate to it, from at least four independent anchor nodes (n_0, n_1, n_2, n_3). Consequently, the non-localized node (n_i) is able to estimate its distances from anchor nodes.
(b) – A standard triangulation can be performed using estimated distances from anchor nodes.

Figure 6. Schematic representation of TERRAIN algorithm.

locally defined system, using the distances to four (or more) already located neighbours.

In order to estimate distances from them, each *regular node* waits until at least four independent ABC algorithms 'propagate' to it from four anchor nodes. At that time a standard trilateration can be performed. In general TERRAIN is more accurate than ABC (Savarese *et al.* 2001). As the number of anchor nodes increases, the accuracy of position estimates improves.

A similar approach is presented by Niculescu and Nath (2001), who propose an *ad hoc* positioning system (APS) where at least three landmarks (with GPS receivers) are assumed to be available. Nodes estimate the distance to these landmarks (that may be multiple hops away) according to the number of hops or the route distance obtained by a distance vector algorithm. Node coordinates can be calculated using the trilateration approach.

3.2.1. Network features

As with the ABC algorithm, the TERRAIN algorithm was developed for both 2D and 3D networks. To be located, each node has to be reached by at least four different ABC algorithms, which start from as many anchor nodes. The availability of four neighbours is a necessity but still may not be sufficient for a node location.

3.2.2. Computational workload

The developed algorithm does not require centralized computation. Each single regular node plays the same role. It

(1) receives ranging and location information from neighbouring nodes;
(2) solves a local optimization problem;
(3) and transmits the obtained results to the neighbouring nodes.

The computational complexity for each node is evidently higher than ABC. The number of computations performed by each node is estimated to be $O(m(n+1))$, where n is the number of neighbours and m the number of the ABC algorithms that has reached the node (Kahaner *et al.* 1988).

3.2.3. Benefits

This method reduces error propagation by the use of anchor nodes and a final refinement process. Compared to ABC, TERRAIN is more accurate (Savarese *et al.* 2001).

3.2.4. Drawbacks

If measurements are corrupted by noise the algorithm can lead to dramatically incorrect nodes displacements. The method is not able to prevent such ambiguities. Because of its incremental nature, the complete graph realization is not guaranteed.

3.2.5. Possible improvements

A first solution to prevent error propagation is to increase the number of anchor nodes. A uniform distribution within the sensor network should guarantee low error accumulation. The price to pay is the *a priori* localization of such nodes. A different kind of approach to improve location accuracy leads to the introduction of an iterative refinement process, where each node uses the range measurements and the most recently computed coordinates of each neighbour to refine its position. This process iterates several times until the locations of all the nodes converge. Average position errors are lower after this refinement and the iterative algorithm, starting from a reasonable graph realization, should reduce the risk of diverging (Savarese *et al.* 2001).

3.3. *Savvides et al. localization algorithm*

This algorithm operates on an *ad hoc* network where a small percentage of nodes know their own position (anchor nodes) (Savvides *et al.* 2001). Before describing the algorithm, we introduce the concept of *beacon* and *unknown nodes*. Nodes with unknown positions are defined as unknown nodes, while localized nodes are called beacons. At the beginning only anchor nodes are considered beacons. Unknown nodes measure their distances from an adequate number of neighbouring beacons, and estimate their positions by performing a numeric optimization. The optimization, known as maximum likelihood, is obtained taking the minimum mean square estimate (MMSE) of an error function (EF), defined as the difference between the measured distances and the estimated Euclidean distances (Kahaner *et al.* 1988):

$$\mathrm{EF} = \frac{\sum_{i=1}^{n}[M_i - E_i]^2}{n} \tag{2}$$

where M_i is the ith inter-node measured distance (e.g. using RSS or ToA approaches), E_i is the ith inter-node Euclidean distance (obtained considering the nodes' estimated positions) and n is the number of neighbouring beacons. This process of estimation is known as atomic multilateration.

Once an unknown node estimates its position, it becomes a beacon and broadcasts its position to other nearby unknown nodes, enabling them to estimate their locations. In general an unknown node will perform an atomic multilateration as soon as it receives information from at least four non-coplanar beacons (three non-aligned beacons in 2D networks). This process, defined as iterative multilateration, incrementally repeats until all the unknown nodes obtain an estimate of their position (see Figure 7).

The algorithm is fully distributed, or alternatively, can be implemented by a single centralized node. In this latter case, the algorithm starts by estimating the position of the unknown node with the maximum number of beacons, using an atomic multilateration to obtain better accuracy and faster convergence. Similarly, when an unknown node estimates its location, it becomes a beacon and this process repeats until the positions of all the nodes (which eventually have four or more neighbouring beacons) are estimated.

3.3.1. *Network features*

This anchor-based algorithm was developed for both 2D and 3D networks. In 3D networks, each node must be connected with at least four non-coplanar beacons. In 2D, it must be connected with at least three non-linear beacons. The presence of four neighbours is then a necessary but not a sufficient condition for a new node localization.

3.3.2. *Computational workload*

The algorithm can work in a distributed or a centralized manner. In both ways, each node needs an atomic multilateration algorithm to be implemented. Computational complexity for each node is estimated to linearly increase with the number of neighbouring beacons. Each node performs $O(n)$ computations, n being the number of neighbours.

3.3.3. *Benefits*

The algorithm is relatively easy to implement and it can be fully distributed.

3.3.4. *Drawbacks*

The algorithm suffers from error accumulation, providing inaccurate positions for nodes far from

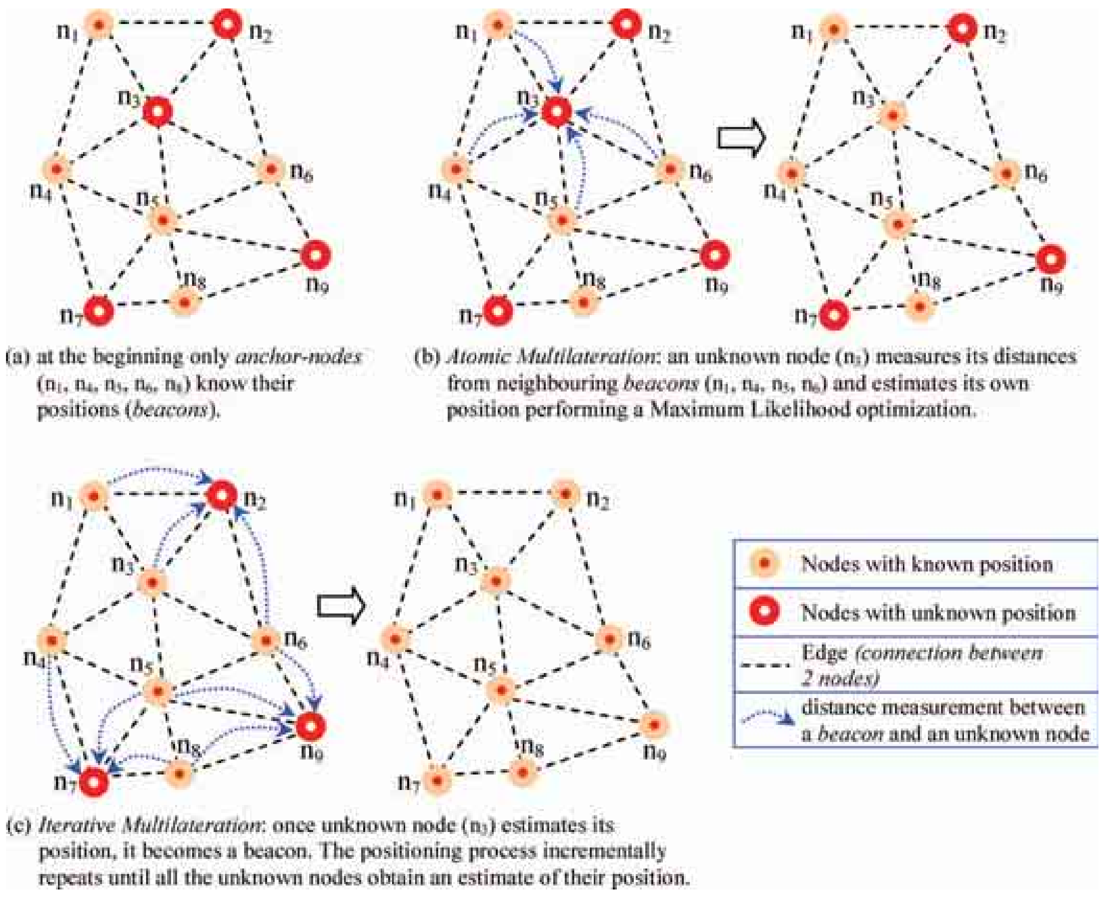

n1
n2
n3
n4
n5
n6
n7
n8
n9
(a) at the beginning only anchor-nodes (n1, n4, n5, n6, n8) know their positions (beacons).
(b) Atomic Multilateration: an unknown node (n3) measures its distances from neighbouring beacons (n1, n4, n5, n6) and estimates its own position performing a Maximum Likelihood optimization.
(c) Iterative Multilateration: once unknown node (n3) estimates its position, it becomes a beacon. The positioning process incrementally repeats until all the unknown nodes obtain an estimate of their position.
Nodes with known position
Nodes with unknown position
Edge (connection between 2 nodes)
distance measurement between a beacon and an unknown node

Figure 7. Schematic representation of Savvides *et al.* localization algorithm (Savvides *et al.* 2001).

anchor nodes. In the centralized version, the error propagation is reduced by first localizing the most connected unknown nodes. Because of its incremental nature, complete graph realization is not guaranteed. If measurements are corrupted by noise, the algorithm can lead to dramatically incorrect nodes displacements.

3.3.5. *Possible improvements*

Error propagation can be minimized through an iterative refinement process – for example, a numerical optimization such as mass–spring relaxation (see section 3.4) – performed after node location.

3.4. *Anchor-free localization (AFL) algorithm*

The anchor-free localization (AFL) algorithm was proposed by Priyantha *et al.* (2003). The algorithm is concurrent, anchor-free, 2D/3D, and proceeds in two phases.

The first phase goal is to produce a qualitative network nodes graph. Arcs are weighted by considering the number of hops. The authors proposed a coarse-grained approach to estimate inter-node distances using hop-count and radio connectivity, without using accurate ranging information from other technologies (e.g. ultrasound). The first phase of AFL can be considered a typical example of coarse-grained algorithm.

For clarity we describe the algorithm for a 2D network. The 3D network case is a simple extension. The algorithm first elects five reference nodes: the first four nodes ($n_1 - n_4$) are selected on the periphery of the graph and the pair $n_1 - n_2$ is roughly perpendicular to the pair nodes $n_3 - n_4$. The remaining node (n_5) is elected in the 'middle' of the graph (see Figure 8). These five nodes are elected in five steps using a hop-count technique based exclusively on radio connectivity.

- *Step 1*. Select an arbitrary node n_0 (see Figure 8). Then, select the reference node n_1 to maximize $h_{0,1}$ (hop-count between nodes n_0 and n_1, i.e. the number of nodes along the shortest radio path between nodes n_0 and n_1).
- *Step 2*. Select reference node n_2 to maximize $h_{1,2}$ (hop-count between nodes n_1 and n_2).
- *Step 3*. Select reference node n_3 to minimize $|h_{1,3} - h_{2,3}|$ and maximize $h_{1,3} + h_{2,3}$. This step selects a node that is roughly equidistant from n_1 and n_2 (1st condition), and is 'far away' from them (2nd condition).
- *Step 4*. As in the previous step, select reference node n_4 to minimize $|h_{1,4} - h_{2,4}|$ and minimize $h_{3,4}$. This optimization selects a node roughly equidistant from nodes n_1 and n_2, while being furthest from node n_3.
- *Step 5*. As in the previous step, select reference node n_5 to minimize $|h_{1,5} - h_{2,5}|$ and maximize $|h_{3,5} - h_{4,5}|$. This optimization selects the node representing the rough 'centre' of the graph.

This heuristic approach uses hop-counts from the chosen reference nodes ($h_{1,i}$, $h_{2,i}$, $h_{3,i}$, $h_{4,i}$, $h_{5,i}$) to determine approximate node coordinates. Further details about the heuristic method can be found in the original paper (Priyantha *et al.* 2003).

The second phase of the AFL algorithm is fine-grained. Inter-node distances are determined using a more accurate measurements technique based on ToA. This is a concurrent phase. Nodes positions are estimated simultaneously by implementing a mass–spring optimization. Nodes are interpreted as concentrated masses, linked by springs. The force that each spring applies to linked nodes depends on the difference between inter-node estimated distances and actual distances (using the ToA method). The starting estimate of inter-node distances is provided by the first phase of AFL. Nodes are gradually moved in order to minimize spring forces providing a more plausible node configuration.

In more detail, each node (n_i) periodically sends its estimated position (p_i) to all its neighbours. Each node

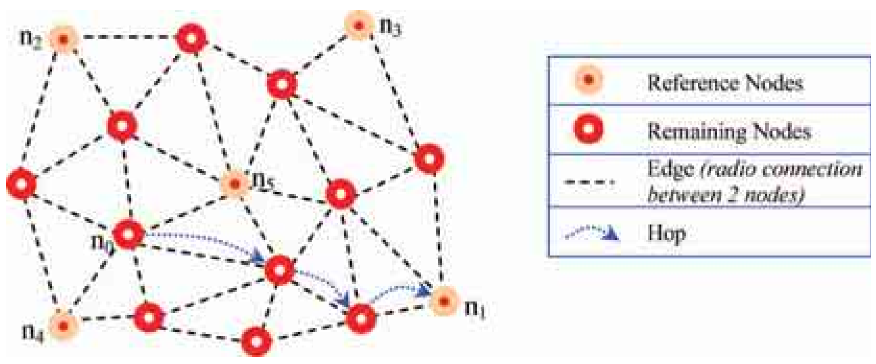


Figure 8. First phase of AFL: election of five reference nodes.

also knows the estimated position of all its neighbours. Using these positions, node n_i calculates the estimated distance ($d_{i,j}$) to each neighbour (n_j). It also knows the distance ($r_{i,j}$), measured using ToA. Let $v_{i,j}$ represent the direction unit vector from p_i (estimated position of ith node) to p_j (estimated position of jth node). The force $F_{i,j}$ along the direction $v_{i,j}$ is given by $F_{i,j} = v_{i,j}(d_{i,j} - r_{i,j})$. The resultant force on the node n_i is given by $F_i = \sum_{j=1}^{N} f_{ij}$, N being the number of neighbours.

The energy $E_{i,j}$ of nodes n_i and n_j (due to the difference in the measured and estimated distances) is directly proportional to the square of $|F_{i,j}|$. The total energy of node n_i is equal to:

$$E_i = \sum_{j=1}^{N} E_{ij} \propto = \sum_{j=1}^{N} (d_{ij} - r_{ij})^2 \tag{3}$$

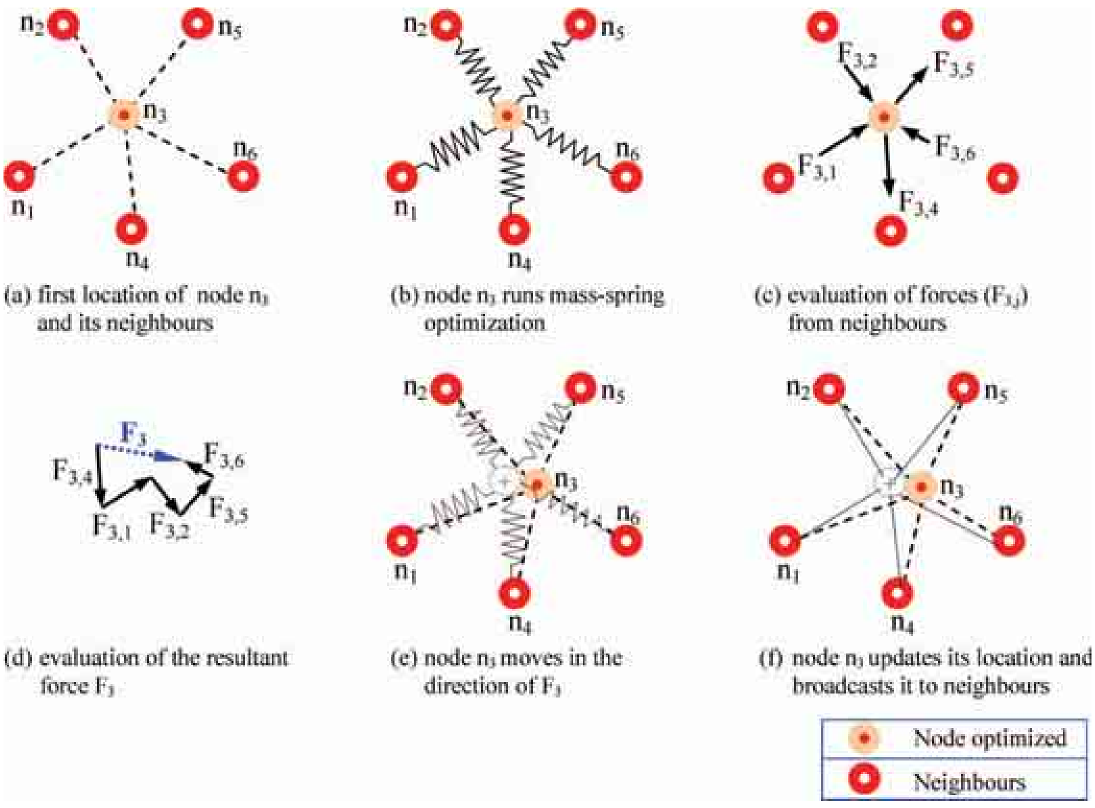
The total energy of the system (E) is given by $E = \sum_{i=1}^{N} E_i$.

In order to reduce its energy E_i, each node n_i moves, one by one, by an infinitesimal amount in the direction of the resultant force F_i. The location of the node is updated and the node broadcasts its new location to its neighbours (see Figure 9). Whenever a node receives a location update from its neighbours, it recalculates its total force and updates its location. The mass–spring optimization terminates when the resultant forces F_i of nodes decrease to zero.

A similar approach was presented by Howard *et al.* (2001). In their system, robots equipped with odometric equipment (instrument indicating the distance travelled) move through an environment, assigning approximate initial positions to beacons. Then, beacons run a distributed spring-based relaxation procedure.

The Gotsman and Koren (2004) algorithm is analogous to AFL. It works in two phases. The first phase produces a qualitative network nodes graph, while the second phase performs an optimization of the network layout.

Wu *et al.* propose a self-configurable positioning technique, quite similar to AFL, built upon two models (Wu *et al.* 2005). First, for a given node distribution, the distance between two nodes (usually multiple hops away) is estimated according to the length of the shortest path. Second, a number of stable nodes are selected to serve as landmarks. Every landmark estimates its distance to other landmarks exchanging obtained distance information. Once a landmark has accumulated a full set of distances between any two landmarks in the network, it may



n2
n5
n3
n6
n1
n4
(a) first location of node n3 and its neighbours
(b) node n3 runs mass-spring optimization
F3,2
F3,5
F3,6
F3,1
F3,4
(c) evaluation of forces (F3,j) from neighbours
F3
F3,4
F3,6
F3,1
F3,2
F3,5
(d) evaluation of the resultant force F3
(e) node n3 moves in the direction of F3
(f) node n3 updates its location and broadcasts it to neighbours
Node optimized
Neighbours

Figure 9. Schematic representation of AFL mass–spring optimization (Priyantha *et al.* 2003).

start establishing the coordinates system by minimizing an error objective function. This latter is defined as the difference between the actual distance and the distance measured in the established coordinates system. Other nodes in the network calculate their coordinates by similarly minimizing the error distances from landmarks.

3.4.1. Network features

AFL is an anchor-free algorithm which applies only to multi-hop networks. In small single-hop networks, where all nodes are connected to each other, it fails. The first phase, based on the hop-count, cannot be executed. AFL applies to both 2D and 3D networks.

3.4.2. Computational workload

The first phase of AFL is far from being distributed. It can hardly be implemented without a centralized network device that handles information from nodes. The second phase of the algorithm is fully distributed, but it can be quite slow since multiple iterations are required (Gotsman and Koren 2004). AFL performances have been evaluated by computer simulations, so it is difficult to provide precise data on the computational workload. AFL is more time-consuming than pure incremental algorithms, due to the number of iterations required. During a single iteration of mass–spring optimization, each node performs $O(n)$ computations, n being the number of neighbours.

3.4.3. Benefits

AFL is anchor-free and does not require nodes with pre-configured coordinates. As opposed to incremental algorithms, AFL performs much better, even for networks with small connectivity (Priyantha *et al.* 2003). Furthermore, AFL error propagation is small.

3.4.4. Drawbacks

The authors do not guarantee that the first phase always succeeds. It may fail for two reasons:

(1) location estimation is extremely rough, especially if the sensor network is composed of few nodes;
(2) in single-hop networks, where all nodes are connected each other, hop-count estimation of inter-node distances does not work.

In general, simulations and practical experiments have demonstrated that a pure mass–spring algorithm can produce networks with incorrect layouts, if initial position estimates are not good (Priyantha *et al.* 2003). The success of the first phase is fundamental for the whole success of the algorithm. Even if AFL outperforms incremental algorithms, there is not a proof of correctness. AFL may converge to distorted network node configurations. If measurements are corrupted by noise, the algorithm can lead to dramatically incorrect node displacements (Savvides *et al.* 2003).

3.4.5. Possible improvements

Present and future improvements are focused on enhancing the first phase. In the actual version, the algorithm lacks a method to prevent realization ambiguities and does not fit widespread networks; as a result it is hardly scalable because of the high communication costs. Research effort focuses on a possible way to realize a completely distributed first phase with such requirements (Gotsman and Koren 2004).

3.5. Moore et al. localization algorithm

Moore *et al.* (2004) proposed a robust distributed algorithm for localizing nodes in a WSN in which measurements are corrupted by noise. In particular, the authors consider how measurement noise can cause incorrect realization of node displacement (see Figure 10). The great benefit of the proposed algorithm is to prevent this ambiguity, increasing positioning accuracy compared to a pure incremental algorithm (Figure 11) (e.g. Savvides *et al.* algorithm).

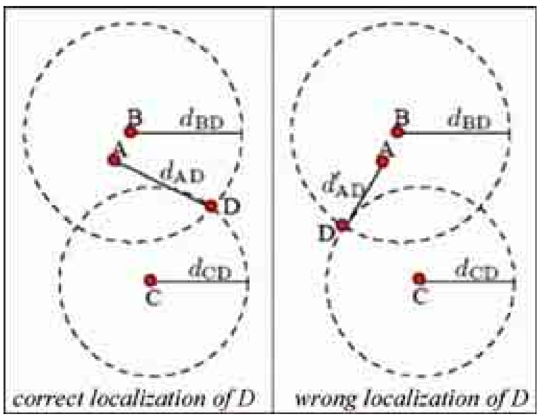


Figure 10. An example of ambiguity: node D is triangulated from the known positions of nodes A, B, and C. Measured distances d_{BD} and d_{CD} constrain the position of D to the two intersections of the dashed circles. Knowing d_{AD} distinguishes between these two positions for D, but a little noise in d_{AD} (shown as d'_{AD}) can lead to a wrong location of node D. Moore *et al.* (2004) provide an algorithm that reduce the probability of such ambiguities.

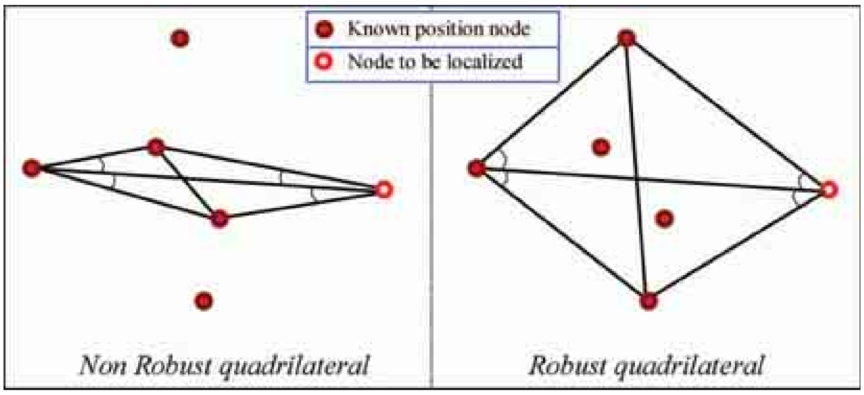


Figure 11. In order to prevent ambiguities such as the one described in Figure 10, localization is performed using robust quadrilaterals. A quadrilateral is defined *robust* if it is regular enough; the idea is that ambiguity occurs using 'flat' quadrilaterals to solve node position (Moore *et al.* 2004).

The algorithm is anchor-free, fine-grained, and it has been physically implemented in 2D sensor networks (Moore *et al.* 2004). Before describing the algorithm, we introduce the concepts of clusters and robust quadrilaterals. A cluster consists of a node and its single-hop neighbours. A robust quadrilateral is an additional constraint that permits localization of only those nodes that have a high likelihood of unambiguous realization. According to Moore *et al.*, localization based on robust quadrilaterals attempts to prevent incorrect realizations of ambiguities.

The algorithm proposed is based on three phases (see Figure 12). In the first phase each node becomes the centre of a cluster and estimates the relative location of neighbours, which can be unambiguously identified. Therefore nodes with ambiguous locations are not used for further node localization. The basic idea of the first phase is that missing localization information for a few probably ambiguous nodes is preferential to estimate incorrect information. This incremental process, called 'cluster localization' is based on trilateration, and 'robust quadrilaterals'.

The second phase is an optional cluster optimization. It refines the position estimates for each cluster using numerical optimization (such as mass–spring relaxation) with the full set of measured distance constraints (see the AFL algorithm). This phase reduces and redistributes any accumulated error that results from the incremental approach used in the first phase. It can be omitted if maximum efficiency is desired.

The third phase computes transformations between the local coordinate systems of neighbouring clusters by finding the set of nodes in common between two clusters and solving for the rotation, translation and possible reflection that best aligns the clusters. This phase is implemented using a 'cluster stitching' technique, presented by Horn (1987). When the third phase is complete, any local cluster coordinate systems are reconciled into a unique global coordinate system (Nagpal *et al.* 2003).

Capkun *et al.* (2001) presented an analogous localization method working with clusters. Each node establishes a local coordinate system for a *cluster*, composed of itself and its one-hop neighbours. Clusters are then stitched together to obtain a coordinate assignment for all the nodes, within a general coordinates system. This technique, unlike that of Moore *et al.*, does not consider how measurement noise can cause incorrect realization of network displacement, and does not prevent this sort of ambiguity (Moore *et al.* 2004).

3.5.1. *Network features*

The localization algorithm is anchor-free. It can be applied to single-hop and multi-hop networks. It is not easily scalable to large networks due to the need for centralized computation in cluster 'stitching'. Until now, it has been implemented only in 2D networks.

3.5.2. *Computational workload*

The first phase of the algorithm is based on trilateration, preceded by non-ambiguity testing. The second phase is a mass–spring relaxation, analogous to the AFL, used to refine the localization of clusters. As a consequence, the algorithm can be quite slow, requiring multiple iterations. These optimizations are performed per cluster and not the network as a whole, thus allowing concurrent processing.

The third phase can hardly be implemented without a centralized network device handling information from clusters that should be stitched together. In this phase, clusters are stitched using a closed-form solution for a least-squares problem. Such a problem

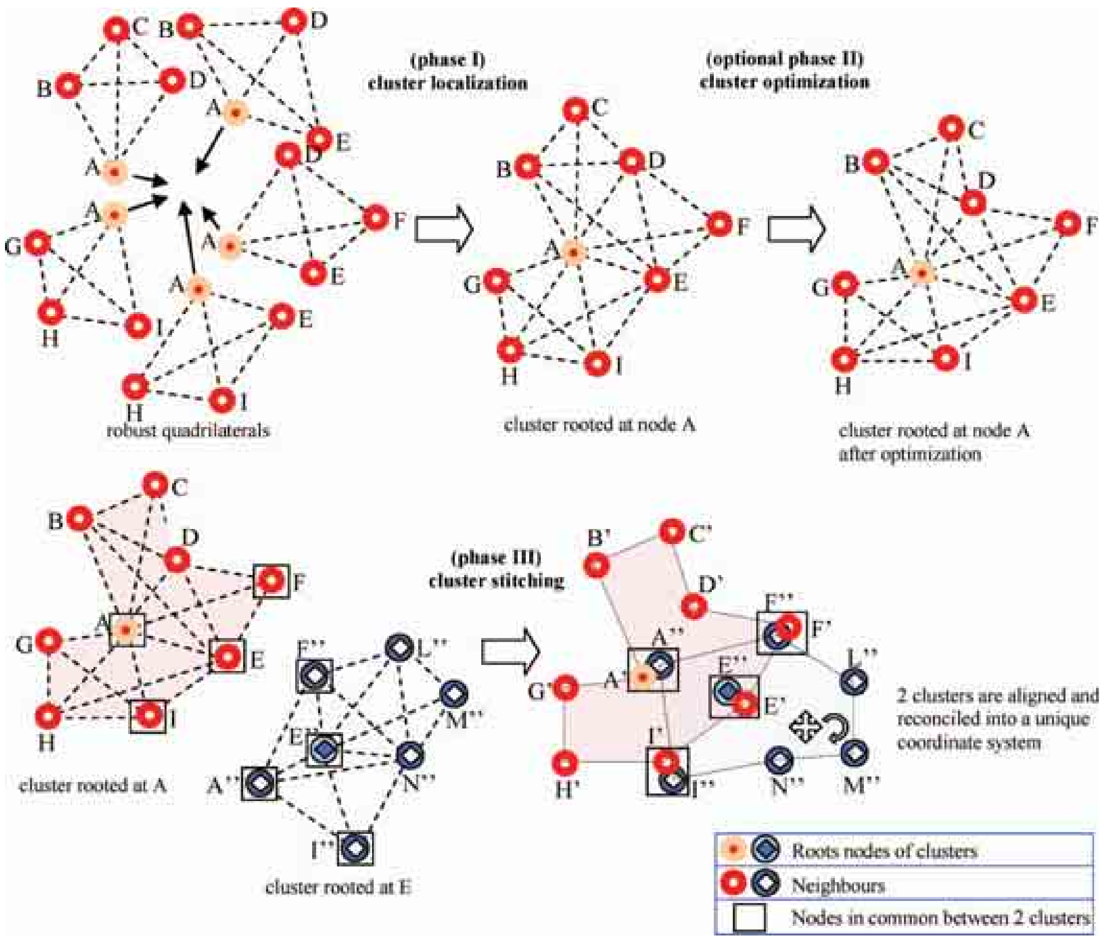


Figure 12. Schematic representation of Moore *et al.* (2004) algorithm.

is relatively complex, with computations required to solve systems of polynomial equations (Horn 1987). The third phase has been exclusively evaluated by computer simulation.

As expected, the computational complexity for each cluster grows with respect to the number of neighbours. For each node, the computation depends on the third power $O(n^3)$ of the number of neighbours n.

3.5.3. Benefits

The algorithm significantly reduces the amount of error propagation over approaches based on basic trilateration. Simulations show that error on node positioning using incremental methods is more than double those using the Moore *et al.* method.

3.5.4. Drawbacks

The drawback of Moore's approach is that under conditions of low node connectivity or high measurement noise, the algorithm may be unable to localize a useful number of nodes. However, for many applications, missing localization information for a known set of nodes is preferential to incorrect information for an unknown set. In 3D networks, computational complexity and data routing dramatically increase.

3.5.5. Possible improvements

When robust quadrilaterals do not exist or when the connectivity is poor, the Moore *et al.* algorithm fails. To get over these difficulties, the algorithm can be enhanced by implementing a more effective robustness test, such as the one proposed by Sottile and Spirito (2006).

4. Summary of localization algorithms

Section 3 provided a detailed description of five significant fine-grained localization algorithms. As discussed earlier, fine-grained algorithms are more accurate than coarse-grained. They utilize more accurate inter-node distances, usually obtained through RSS or ToA techniques. These algorithms are suitable for applications where nodes are required to be localized with a fair level of accuracy. In object tracking, for example, accurate localizations of network nodes lead to accurate locating of objects moving within the network. On the other hand, coarse-grained algorithms provide a rougher localization of nodes, but they are simpler to implement.

In this section, the five localization algorithms are compared according to the taxonomy presented in Table 1. Identified criteria can be useful to evaluate and compare different network localization techniques. The aim is to provide a reference scheme to select them, depending on network characteristics (see Table 2). Considering the actual research issues related to localization algorithms, there is much room for improvement. Several researchers are trying to develop existing algorithms in order to make them work in non-optimal conditions (for example incomplete connectivity, presence of moving sensors) (Taylor *et al.* 2005, Sottile and Spirito 2006). Additional effort is being spent to bridge the gap between simulations and real-world localization systems by gathering more data on the real behaviour of sensor nodes, particularly with respect to physical effects like multipaths, interference and obstruction (Langendoen and Reijers 2003). Furthermore, other research groups are studying the problem of 'directional localization', where each network node not only must be aware its position but also its orientation relative to the network (Akcan *et al.* 2006).

5. Conclusions

In many applications of WSNs, it is crucial to determine the physical location of nodes. Automatic localization of nodes in wireless networks is a key to enable most of these applications. As an example, we considered a sensor network deployment within a warehouse. Making sensors wireless and self-configurable reduces the high cost of cabling and makes the network more manageable and dynamic.

Numerous network localization algorithms have been recently proposed and developed by many authors. Similarities are present in each approach (Langendoen and Reijers 2003, Patwari *et al.* 2005). This paper suggests a new taxonomy to help evaluate, compare and select network localization algorithms, depending on the network characteristics and the type of applications.

The paper focused on five fine-grained techniques, due to their better accuracy and their better chances of being applied to many contexts (e.g. quality control, indoor navigation, logistics, warehousing, remote diagnostics, etc.). Algorithms have been discussed in detail in order to summarize their characteristics and peculiarities.

Many algorithms have never been tested in practice. Additional effort is needed to test algorithms with practical experiments, and not only through simulations, in order to assess their performance and reliability. Various algorithms are in testing on two specific applications at the industrial metrology and quality laboratory of DISPEA, Politecnico di Torino:

(1) innovative techniques for taking coordinate dimensional measurements of objects, using distributed wireless sensors (Franceschini *et al.* 2002);
(2) wireless monitoring of systems with changeable configuration (e.g. cranes, mechanical arms, automatic gates, etc.) to check their 'natural' positions.

Since these applications require a reasonable level of accuracy in inter-node distance estimates, network nodes are equipped with ultrasound transceivers implementing a ToA technique.

References

Akcan, H., Kriakov, V., Brönnimann, H., and Delis, A., 2006. GPS Free node localization in mobile wireless sensor networks. *In*: *Proceedings of MobiDE'06*, 35–42.

Bulusu, N., Heidemann, J., and Estrin, D., 2000. GPS-less low cost outdoor localization for very small devices. *IEEE Trans. Pers. Commun. Mag.*, 7, 28–34.

Capkun, S., Hamdi, M., and Hubaux, J.P., 2001. GPS-free positioning in mobile ad-hoc networks. *In*: *Proceedings of the 34th Hawaii International Conference on System Sciences*, 9008–9018.

Chen, M., Cheng, F., and Gudavalli, R., 2003. *Precision and Accuracy in an Indoor Localization System*, University of California, Berkeley, Technical Report CS294-1/2.

Doherty, L., Pister, K.S.J., and Ghaoui, L.E., 2001. Convex position estimation in wireless sensor networks. *In*: *Proceedings of IEEE INFOCOM*, 1655–1663.

Doss, R.C. and Chandra, D., 2005. Reliable event transfer in wireless sensor networks deployed for emergency response. *In*: *Proceedings of the 17th IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2005)*, 208–213.

Franceschini, F., Galetto, M., Maisano, M., and Mastrogiacomo, L., 2006. *Mobile Spatial Coordinate Measuring System (MScMS) – Introduction to the System*, Work in progress, DISPEA, Politecnico di Torino, Italy.

Franceschini, F., Galetto, M., and Settineri, L., 2002. On-line diagnostic tools for CMM performance. *Int. J. Adv. Manufact. Technol.*, 19, 125–130.

Gotsman, C. and Koren, Y., 2004. Distributed graph layout for sensor networks. *In*: *Proceedings of Graph Drawing 12th International Symposium*, 273–284.

Horn, B.K.P., 1987. Closed form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am.*, 4, 629–642.

Howard, A., Mataric, M., and Sukhatme, G., 2001. Relaxation on a mesh: a formalism for generalized localization. *In*: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1055–1060.

Intel Corporation, Expanding usage models for wireless sensor networks, 2005. Available online at Technology@ Intel Magazine, http://www.intel.com/technology/magazine/research/sensor-networks-0805.pdf

Ji, X. and Zha, H., 2004. Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling. *In*: *Proceedings of IEEE INFOCOM*, 2652–2661.

Kahaner, D., Moler, C., and Nash, S., 1988. *Numerical Methods and Software*. Englewood Cliffs, NJ: Prentice-Hall.

Koumpis, K., Hanna, L., Andersson, M., and Johansson, M., 2005. Wireless industrial control and monitoring beyond cable replacement. *In*: *Proceedings of PROFIBUS International Conference*, paper C1, 1–7.

Langendoen, K. and Reijers, N., 2003. Distributed localization in wireless sensor networks: a quantitative comparison. *Comput. Netw.*, 43, 499–518.

Moore, D., Leonard, J., Rus, D., and Teller, S.S., 2004. Robust distributed network localization with noisy range measurements. *In*: *Proceedings of SenSys 2004*, 50–61.

Nagpal, R., Shrobe, H., and Bachrach, J., 2003. Organizing a global coordinate system from local information on an ad hoc sensor network. *In*: *Proceedings of International Workshop on Information Processing in Sensor Networks (IPSN 2003)*, 267–280.

Nasipuri, A. and Li, K., 2002. A directionality based location discovery scheme for wireless sensor networks. *In*: *Proceedings of ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, 105–111.

Niculescu, D. and Nath, B., 2001. Ad hoc positioning system (APS). *In*: *Proceedings of IEEE Global Communications Conference (GlobeCom'01)*, 2926–2931.

Niculescu, D. and Nath, B., 2003. Ad hoc positioning system (APS) using AOA. *In*: *Proceedings of IEEE Annual Joint Conference IEEE Computer and Communications Societies (INFOCOM'03)*, 1734–1743.

Oh, S., Chen, P., Manzo, M., and Sastry, S., 2006. Instrumenting wireless sensor networks for real-time surveillance. *In*: *Proceedings of the International Conference on Robotics and Automation*, 3128–3133.

Pan, M., Tsai, C., and Tseng, Y., 2006. Emergency guiding and monitoring applications in indoor 3D environments by wireless sensor networks. Available online at: http://www.cs.berkeley.edu/~kamin/pubs/icra06_mttvideo.pdf

Patwari, N., Ash, J., Kyperountas, S., Hero III, A., Moses, R., and Correal, N., 2005. Locating the nodes – cooperative localization in wireless sensor networks. *IEEE Sig. Proc. Mag.*, 22, 54–69.

Pepperl + Fuchs, 2005. *Internal Report on Factory Automation*. Available online at: http://www.pepperl-fuchs.com

Priyantha, N.B., Balakrishnan, H., Demaine, E., and Teller, S., 2003. *Anchor-free Distributed Localization in Sensor Networks*, MIT Computer Science Lab., Technical Report 892.

Savarese, C. and Rabaey, J., 2002. Robust positioning algorithms for distributed ad hoc wireless sensor networks. *In*: *Proceedings of USENIX Annual Technical Conference*, 317–327.

Savarese, C., Rabaey, J., and Beutel, J., 2001. Locationing in distributed ad-hoc wireless sensor networks. *In*: *Proceedings of ICASSP*, 2037–2040.

Savvides, A., Garber, W., Adlakha, S., Moses, R., and Strivastava, M.B., 2003. On the error characteristics of multihop node localization in ad-hoc sensor networks. *In*: *Proceedings of IPSN*, 317–332.

Savvides, A., Han, C., and Strivastava, M.B., 2001. Dynamic fine-grained localization in ad hoc networks of sensors. *In*: *Proceedings of ACM/IEEE 7th Annual International Conference on Mobile Computing and Networking (MobiCom'01)*, 166–179.

Sottile, F. and Spirito, M., 2006. Enhanced quadrilateral-based localization for wireless ad-hoc networks. *In*: *Proceedings of IFIP 5th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2006)*, 224–231.

Taylor, C., Rahimi, A., Bachrach, J., and Shrobe, H., 2005. *Simultaneous Localization, Calibration, and Tracking in an ad Hoc Sensor Network*, Computer Science and Artificial Intelligence Laboratory of MIT, Technical Report. Available online at: https://dspace.mit.edu/handle/1721.1/30541

Wang, L. and Xi, F., 2006. Challenges in design and manufacturing. *Int. J. Comp. Int. Manufact*, 19, 409–410.

Ward, A., Jones, A., and Hopper, A., 1997. A new location technique for the active office. *IEEE Trans. Pers. Commun.*, 4, 42–47.

Wu, H., Wang, C., and Tzeng, N., 2005. Novel self-configurable positioning technique for multihop wireless networks. *IEEE/ACM Trans. Network*, 3, 609–621.