Transaction on Wireless Communication
ISSN: 2229-8711 Online Publication, June 2011
www.pcoglobal.com/gjto.htm
NS-N11 /GJTO

Global Journal of Technology & Optimization
Volume 2, 2011

SPACE AND TIME IN WIRELESS SENSOR NETWORKS

Ustijana R. Shikoska, Danco Davcev, Risto Reckoski, Gordana Petrovska R., Cvetko Andreevski, Jordan Sikoski

University for Information Sciences & Technologies, "Sv. Apostol Pavle", Ohrid, Republic of Macedonia
Sv. Kiril i Metodij" University, Faculty of Electrical Engineering, Skopje, Republic of Macedonia
Sv. Kliment Ohridski" University, Faculty for Tourism and Hospitality, Ohrid, Republic of Macedonia
University of American Colledge, Skopje, Republic of Macedonia
Email: ustijana@t-home.mk

Received January 2011, Revised February 2011, Accepted March 2011

Abstract
Wireless sensor networks (WSN) are an increasingly attractive means to bridge the gap between the physical and virtual world. *WSNs* are envisioned to be used to fulfill complex monitoring tasks. Space and time play a crucial role in wireless sensor networks, since sensor nodes are used to collaboratively monitor physical phenomena and their space-time properties. A number of techniques and distributed algorithms for location estimation and time synchronization have been developed specifically for sensor networks. Time synchronization is a crucial component of *WSNs*. Time synchronization schemes developed for traditional networks are ill-suited for *WSNs* and more appropriate approaches should be suggested. There are many similarities in space and time domains. This affects the location estimation and time synchronization, ranging from applications and requirements to basic approaches and concrete algorithmic techniques. The purpose of this paper is to make this close affinity explicit in order to further a better understanding and improving both domains.

Keywords: space localization, time synchronization, wireless sensor networks, localization algorithms

1. Introduction

Enabled by technological advancements in wireless communications and embedded computing, wireless sensor networks were first considered for military applications, where large-scale wireless networks of autonomous sensor nodes would enable the unobtrusive observation of events in the real-world. Since then, the use of sensor networks has also been considered for various civil application domains.

Having observed the speed of technological advancements over the past, one could envision at that time that in the near future it would be possible to build even smaller untethered computing, communicating, and sensing devices with marginal cost per device. While the low per-device cost would allow mass production, small size would enable an unobtrusive deployment.

This prospect triggered researchers to think of implications and applications of this emerging new technology. This vision was further refined and substantiated by a number of visionaries and research projects. This development was evidenced by several new terms such as Pervasive Computing, Ambient Intelligence, also Wireless Sensor Networks (*WSN*).

Common to these slightly different terms and underlying visions is the goal of bridging the long standing gap between the physical world and the traditional virtual world of computers and other information-technology artifacts.

The key to realization of these visions is the use of large collections of these unobtrusive networked computers that could perceive and control aspects of the real world via sensors and actuators on the one hand, and that would provide an intuitive interface to human users on the other hand. While projects classified as Ubiquitous Computing, Pervasive Computing, and Ambient Intelligence are somewhat focused on issues related to interfacing these unobtrusive networked computing devices to human users, this component is of lesser significance in projects that examine sensor networks. Research on wireless sensor networks focuses on technical aspects of observing the real world with best possible quality, using as few as possible resources, and minimizing the impact of the observation tool on the observed physical processes.

WSN have been initially considered for military applications, where real-world events (e.g., vehicles and troops passing) must be unobtrusively observed in inaccessible or hostile environments. For example, *DARPA* initiated the Distributed Sensor Networks program in the *1980*-ies. For these military tasks, large numbers of sensor nodes would be deployed in the area of interest and form a wireless network to observe events in the physical environment. These long-lived, unattended networks would be unobtrusive due to the small size of individual nodes,

could operate without the use of additional hardware infrastructure and would be robust due to the redundant deployment of nodes. Later on, it was suggested that these features would render sensor networks a useful tool also in a number of civil application domains, for example as a scientific tool for environmental monitoring or in building automation.

Time and *space* are fundamental categories in the physical world. Since *wireless sensor networks* are a tool for observing, influencing, and reasoning about phenomena of the physical word, time and space are also of utmost importance in *WSN*. They are essential elements for obtaining and interpreting real-world observations (e.g., where and when did an event occur, how large and fast was an observed object), for tasking a sensor network (e.g., where and when to look for events), for interfacing wireless sensor networks with the real-world (e.g., what node density and sampling frequency is needed to observe a certain object), and for coordination among sensor nodes (e.g., which nodes can when be switched to idle mode). There are two basic services to enable these functions: *time synchronization and localization of sensor nodes*. Time synchronization allows a sensor node to estimate current time with respect to a common time scale. Localization allows a node to estimate its current location with respect to a common coordinate system.

The categories time and location are fundamental for many applications of sensor networks, due to the close integration of sensor networks with the real world. Interpretation of sensing results or coordination among sensor nodes are some of the implementations, *time synchronization* and *sensor node localization* are fundamental and closely related services in sensor networks.

Existing solutions were based on a rather narrow notion of a sensor network as a large-scale, ad hoc, multi-hop, un-partitioned network of largely homogeneous, tiny, resource-constrained, mostly immobile sensor nodes that would be randomly deployed in the area of interest. Recently developed prototypical applications indicate that this narrow definition does not cover a significant portion of the application domain of wireless sensor networks.

Applications of sensor networks span a whole design space with many important dimensions. Existing solutions for time synchronization and node localization do not cover all important parts of the design space. Different approaches are required to support these regions adequately. Such solutions can be provided.

2. Related work

There are different ways of classifying general *space and time localization algorithms,* they can be classified according to the measurement assumptions as four types: 1) connectivity-only 2) range-based 3) angle-based 4) hybrid. A comparison between the more well - known algorithms such as *DV-Hop (Distance and Euclidean)*, Euclidean and Multi-lateralization can be obtained from [12]. The comparison is done in the context of specific constraints of sensor networks, such as error tolerance and energy efficiency, results indicate that there is no single algorithm that performs "best" and that there is possibility for further improvement.

A number of localization methods rely on connectivity information only. These types of methods are also referred to as "range-free" methods. The *Centroid method* [13] estimates the location of an unknown node as the average of its neighbors' locations. The *APIT* method *(Ad Hoc Positioning* [14] estimates the node location by isolating the area using various triangles formed by beacons. The *DV-Hop method* [15] counts the hop numbers to beacons and uses them as crude estimates for distances. Range-free methods require no additional hardware, but they generally only work well when networks are dense. Sparse networks by nature contain less connectivity information, and thus they are more difficult to localize accurately.

Range-based methods include the *Ad Hoc Positioning System (APS*) methods such as *DV-Distance and Euclidean* proposed in [15, 16]. In [17], ranging data are exchanged between the neighbors to refine the initial location guess. While those methods compute the absolute node locations, the *GPS-Free method* [18] calculates the relative node locations from the distance measurements. Compared to range-free methods, range-based methods give more accurate location estimates when ranging data is reliable. However, depending on the deployment environment, ranging techniques based on *RSSI-Received Signal Strength Indicator* tend to be error-prone and strong filtering is required. The ranging error could ultimately destroy the localization accuracy if it is allowed to propagate through the network unbounded.

Different methods generally exploit the trade-off between the estimation accuracy and the estimation coverage. For instance, given the same network scenario, the Euclidean method is capable of generating more accurate location estimates of a smaller subset of nodes, whereas the *DV-Hop* method has better coverage but worse accuracy. Regardless of the tradeoff, a common characteristic shared by distance-based *algorithm* is that they require a relatively high network density in order to achieve better results. Based on the extensive simulation of *DV-Distance, Euclidean* and *multilateration methods* performed in [19], it can be concluded that those distance-based *GAHLA*s "require an average degree of *11-12* nodes within the ranging neighborhood in order to achieve *90%* localization coverage with *5%* accuracy [19]."

Even though the future of AoA sensing devices is still unclear, some works have been published on localization using angle information. Simulation studies in [19] also show that when AoA (angle of arrival) of the signals is used in addition to the distance measurement, the localization accuracy and coverage can be drastically improved.

A combination of the above techniques can be employed to form a hybrid method. For instance, a hybrid method is proposed in [20] that uses both *APS* and *MDS* (multidimensional scaling).

The goal of the localization algorithm is to shape the distribution based on a sequence of measurement until the distribution becomes focused and collapses onto a small area. The probabilistic method and particle filters have been used in visual target tracking and computer vision location systems [21] in the context of robotics. The particle filter method is also used to obtain the mobile node location based on received signal strengths from several known-location base stations in wireless cellular networks. The probability grid system in is a centralized probabilistic localization algorithm that updates the distribution based on a grid system.

The indoor location tracking problem in deals with a known environment thus different obstacles can be represented in a floor-plan and thus a signal strength *(RSSI*) map can be obtained via measurements and calculations ahead of time. The location tracking problem then becomes a decision-making problem, where a solution may use a measurement model that compares the current *RSSI* with the signal strength map to find the location with the highest probability of matching the current *RSSI* reading. While roots are similar, the solution described here is designed for out-door environments and infrastructure-less networks where major continuous obstacles are assumed to be minimum, and fairly reliable distance estimates can be obtained from *RSSI* readings and the signal propagation model. The probability distributions of location estimates are updated solely from the distance and location estimates from neighbors.

3. Wireless Sensor Networks

Research on wireless sensor networks goes back to a number of research projects, where the use of large networks of tiny wireless sensor devices was explored in a military domain. Initial work mainly focused on the development of hardware prototypes and energy-efficient networking protocols. These early efforts established a definition of a wireless sensor network as a large-scale, wireless, ad hoc, multi-hop network of homogeneous, tiny, mostly immobile sensor nodes that would be randomly deployed in the area of interest. Since then, the use of wireless sensor networks has also been considered for a number of civil applications. Wireless sensor networks have been suggested as a scientific tool for better understanding real-world phenomena, as an enabling technology for making our daily life more comfortable, as a tool for improving the efficiency of industrial processes, and as a mechanism for dealing with issues such as environmental protection and law enforcement. In these application domains, wireless sensor networks are deemed a promising technology with the potential for changing the way of living by bridging the gap between the real world and the virtual world of existing information technology.

Sensor networks consist of sensor nodes, computing devices that include a power source, a transceiver for wireless communication, a processor, memory, sensors, and potentially also actuators. Although the exact properties and capabilities of these components may vary, a common property of sensor nodes is their resource scarcity.

Multiple sensor nodes form a wireless network, whose topology and other properties do also depend on the application context. A large class of sensor networks can be characterized as multi-hop ad hoc networks, where sensor nodes do not only act as data sources, but also as routers that forward messages on behalf of other nodes, such that no additional communication infrastructure is required for operating the network.

The sensor nodes participating in a network can vary in their capabilities and configuration. Sensor nodes may be equipped with different types of sensors; some sensor nodes might be equipped with a more powerful processor and more memory to perform sophisticated computations; some nodes might be connected to a other networks and can act as gateways to a background infrastructure. Using attached sensors, nodes can observe a partial state of the real world in their close physical neighborhood. By integrating observations of many sensor nodes, a more detailed and geographically extensive observation of a partial state of the real world can be obtained. Due to the relatively small effective range of sensor nodes, sensor networks often consist of many, densely deployed sensor nodes.

While individual sensor nodes have only limited functionality, the global behavior of a sensor network can be quite complex. The true value of the network is in this emergent behavior: the functionality of the whole is greater than the sum of its parts. A sensor network may estimate the velocity of a moving object even though sensor nodes are not equipped with velocity sensors. Instead, velocity estimates can be obtained by correlating object sightings from spatially dispersed sensor nodes, which requires only sensors for detecting the proximity of objects.

Wireless communication, especially with focus on short communication range and low power consumption, is a key enabling technology for wireless sensor networks. In mobile networks, computers capable of wireless communication can change their physical position over time, resulting in dynamically changing network topologies. Ad hoc networks are wireless networks that do not require an external infrastructure such as base stations in mobile phone networks. The nodes of an ad hoc network act both as sources/sinks of messages and as routers that forward messages on behalf of other nodes. Nodes can join and leave the network anytime. Although ad hoc networks may also consist of immobile nodes, they often contain mobile nodes. Power awareness is an important issue in the context of mobile networks, since mobile computing devices are often powered by batteries. Recent research in mobile ad hoc networks focuses on routing, mobility management, power management, self-configuration, and the radio interface (including the radio hardware and medium access techniques). Many wireless sensor networks will be implemented as a mobile ad hoc network (MANET). However, results from MANET research often cannot be directly applied to wireless sensor networks, since resource and energy constraints are typically more stringent here. Typical MANET research focuses on handheld devices or laptops with renewable batteries. The computing, storage, communication resources of these devices are comparable to desktop computers. In contrast, sensor node batteries are often not replaceable; range, bandwidth, reliability of wireless communication links, computing and memory resources, and available energy may be orders of magnitude smaller compared to more traditional MANET nodes.

The deployment of sensor nodes in the physical environment may take several forms. Nodes may be deployed at random (e.g., by dropping them from an aircraft) or installed at deliberately chosen spots. Deployment may be a one-time activity, where the installation and use of a sensor network are strictly separate activities. However, deployment may also be a continuous process, with more nodes being deployed at any time during the use of the network – for example, to replace failed nodes or to improve coverage at certain interesting locations. The actual type of deployment affects important properties such as the expected node density, node locations, regular patterns in node locations, and the expected degree of network dynamics.

Sensor nodes may change their location after initial deployment. Mobility can result from environmental influences such as wind

or water, sensor nodes may be attached to or carried by mobile entities, and sensor nodes may possess automotive capabilities. Mobility may be either an incidental side effect, or it may be a desired property of the system (e.g., to move nodes to interesting physical locations), in which case mobility may be either active (i.e., automotive) or passive (e.g., attached to a moving object not under the control of the sensor node). Mobility may apply to all nodes within a network or only to subsets of nodes. The degree of mobility may also vary from occasional movement with long periods of immobility in between, to constant travel. Mobility has a large impact on the expected degree of network dynamics and hence influences the design of networking protocols and distributed algorithms. The actual speed of movement may also have an impact, for example on the amount of time during which nodes stay within communication range of each other.

Wireless sensor networks may also rely on infrastructure-based mobile networks. For example, mobile phone companies are currently exploring the value of mobile phones for sensor networks. Such networks could either solely consist of mobile phones equipped with sensors, or a mobile phone could act as a gateway connecting an ad hoc sensor network to the phone network. Such combinations of infrastructure-based and ad hoc networks would allow remote access to sensor networks and an integration with existing computing infrastructures.

Early sensor network visions anticipated that sensor networks would typically consist of homogeneous devices that were mostly identical from a hardware and software point of view. Some projects assumed that sensor nodes were indistinguishable, that is, they did not even possess unique addresses within their hardware. This view was based on the observation that otherwise it would not be feasible to cheaply produce vast quantities of sensor nodes. However, in many prototypical systems available today, sensor networks consist of a variety of different devices. Nodes may differ in the type and number of attached sensors; some computationally more powerful “compute” nodes may collect, process, and route sensory data from many more limited sensing nodes; some sensor nodes may be equipped with special hardware such as a *GPS* receiver to act as beacons for other nodes to infer their location; some nodes may act as gateways to long-range data communication networks (e.g., *GSM* networks, satellite networks, or the Internet). The degree of heterogeneity in a sensor network is an important factor since it affects the complexity of the software executed on the sensor nodes and also the management of the whole system.

The various communication modalities can be used in different ways to construct an actual communication network. Two common forms are so-called infrastructure based networks on the one hand and ad hoc networks on the other hand. In infrastructure-based networks, sensor nodes can only directly communicate with so-called base station devices. Communication between sensor nodes is relayed via the base station. If there are multiple base stations, these have to be able to communicate with each other. The number of base stations depends on the communication range and the area covered by the sensor nodes. Mobile phone networks are an example of this type of network.

In ad hoc networks, nodes can directly communicate with each other without an infrastructure. Nodes may act as routers, forwarding messages over multiple hops on behalf of other nodes. Since the deployment of an infrastructure is a costly process, and the installation of an infrastructure may often not be feasible, ad hoc networks are preferred for many applications. However, if an infrastructure is already available anyway, it might also be used for certain sensor network applications. Combinations of ad hoc networks and infrastructure-based networks are sometimes used, where clusters of sensor nodes are interconnected by a wide area infrastructure-based network. Note that the above arguments not only apply to communication, but also to other infrastructures, such as localization or time synchronization.

One important property of a sensor network is its diameter, that is, the maximum number of hops between any two nodes in the network. In its simplest form, a sensor network forms a single-hop network, with every sensor node being able to directly communicate with every other node. An infrastructure-based network with a single base station forms a star network with a diameter of two. A multi-hop network may form an arbitrary graph, but often an overlay network with a simpler structure is constructed such as a tree or a set of connected stars. The topology affects many network characteristics such as latency, robustness, and capacity. The complexity of data routing and processing also depends on the topology.

The communication ranges and physical locations of individual sensor nodes define the connectivity of a network. If there is always a network connection (possibly over multiple hops) between any two nodes, the network is said to be connected. Connectivity is intermittent if the network may be occasionally partitioned. If nodes are isolated most of the time and enter the communication range of other nodes only occasionally, communication is sporadic. Note that despite the existence of partitions, messages may be transported across partitions by mobile nodes. Connectivity mainly influences the design of communication protocols and methods of data gathering.

The number of nodes participating in a sensor network is mainly determined by requirements relating to network connectivity and coverage, and by the size of the area of interest. The network size may vary from a few nodes to thousands of sensor nodes or even more. The network size determines the scalability requirements with regard to protocols and algorithms.

Depending on the application, the required lifetime of a sensor network may range from some hours to several years. The necessary lifetime has a high impact on the required degree of energy efficiency and robustness of the nodes.

Depending on the application, a sensor network must support certain quality of service aspects such as real-time constraints (e.g., a physical event must be reported within a certain period of time), robustness (i.e., the network should remain operational even if certain well-defined failures occur), tamper-resistance (i.e., the network should remain operational even when subject to deliberate attacks), eavesdropping-resistance (i.e., external entities cannot eavesdrop on data traffic), unobtrusiveness or stealth (i.e., the presence of the network must be hard to detect).

The output of the sensor network may be used for various purposes. Thee output is delivered to a human user for further evaluation. It may be used to control the operation of the sensor network without human intervention by enabling/disabling sensors, or by controlling operation parameters of sensors (e.g., sampling rate, sensitivity, orientation, position). Using the output

of the sensor network to control sensors or actuators can effectively create a closed-loop system that strives to achieve a particular nominal condition in the sensor network or in the real world. There is a growing trend to instrument cars with more and more sensors and actuators to improve the drivability and comfort. While past research mostly focused on single cars, recent developments include networking of cars [8], [10] to reduce accidents, traffic jams, environmental stress, or to improve fleet management.

The system can help find streets in the locality with vacant spots, can find occupied parking meters within a certain range which will expire at a certain time, and can locate all vehicles that reside in expired spots. In this system, parking meters are equipped with sensor nodes. These nodes are equipped with sensors to detect the occupancy status of the according parking spot and have access to parameters of the parking meter such as time of expiry. The sensor nodes form a static multi-hop ad hoc network. Cars are also equipped with sensor nodes that establish a link to the meter network to issue queries about free parking spots.

The characteristics of wireless sensor networks can present a number of major challenges to the development of algorithms, protocols, and systems. The main technical challenges are resource and energy constraints, network dynamics, network size and density, unattended and un-tethered operation.

Wireless sensor networks can be considered as a tool for observing real-world processes. In particular, the use of WSN might be a worthwhile option for observation tasks with one or more of some properties: the observation environment is cluttered and can be hardly observed from afar; any instrumentation for observation must be unobtrusive to avoid influencing observation results; the phenomenon of interest or its close physical environment can be instrumented for observation; a high spatial and temporal monitoring resolution is required; the signal-to-noise ratio of signals emitted by the phenomenon of interest is low or decreases significantly over distance; the observation environment is very harsh, inaccessible or even toxic; the observation must be continuously performed during long periods of time or over large geographical areas. Depending on the actual needs of the application, the form factor of a single sensor node may vary from the size of a shoe box (e.g., a weather station) to a microscopically small particle (e.g., for military applications where sensor nodes should be almost invisible). Similarly, the cost of a single device may vary from hundreds of Euros (for networks of very few, but powerful nodes) to a few Cents (for large-scale networks made up of very simple nodes). Since sensor nodes are autonomous devices, their energy and other resources are limited by size and cost constraints. Varying size and cost constraints directly result in corresponding varying limits on the energy available (i.e., size, cost, and energy density of batteries or devices for energy scavenging), as well as on computing, storage, and communication resources.

Hence, the energy and other resources available on a sensor node may also vary greatly from system to system. Power may be either stored (e.g., in batteries) or scavenged from the environment (e.g., by solar cells). These resource constraints limit the complexity of the software executed on sensor nodes. It is important to ensure that resource usage and energy consumption is equally spread among the nodes of the network. If some nodes exhaust their battery quickly and fail early, resulting permanent network partitions may render the network in-operational. Usage of resources may lead to bottlenecks such as network congestions. Sensor nodes send sensor readings along a spanning tree to a base station for evaluation. Nodes close to the base station will run out of power since they forward messages from nodes further away.

Depleted batteries and corruptive environmental conditions often lead to node failures. Temporary environmental obstructions (e.g. vehicles, humans) may influence the communication range of nodes. Nodes may be mobile, new nodes may be added to replace failed ones. All these issues may lead to frequent topology changes in sensor networks. Temporary network partitions are likely to exist in sparse networks.

Despite intermittent connectivity, messages can be forwarded across partitions by mobile nodes as illustrated in Figure *1*.

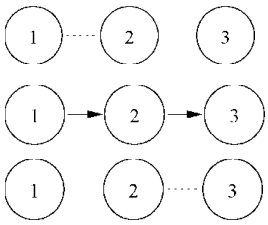


Fig.1: Message transport across partition boundaries through node mobility

At time t_1 nodes *1* and *2* are within communication range of each other only, then node *2* moves towards node *3,* such that at time t_2 nodes *2* and *3* are within communication range of each other only. Node *2* can carry a message from node *1* to node *3* across a partition boundary. The resulting multi-hop message flow has two challenging properties. The path is unidirectional: it is not possible to send a message from node *3* to node *1* unless there is a node with an appropriate mobility pattern. The delay of this message flow can be arbitrarily high and is hardly predictable unless the mobility pattern of node *2* is known in advance.

One important property of a sensor network is its diameter, that is, the maximum number of hops between any two nodes in the network. In its simplest form, a sensor network forms a single-hop network, with every sensor node being able to directly communicate with every other node.

An infrastructure-based network with a single base station forms a star network with a diameter of two. A multi-hop network may form an arbitrary graph, but often an overlay network with a simpler structure is constructed such as a tree or a set of connected stars. The topology affects many network characteristics, such as latency robustness, and capacity. The complexity of data routing and processing also depends on the topology.

Ensuring robust operation of a sensor network in such setups can be a very challenging task.

4. Space and Time in Sensor Networks

The close relationship between time and space in the physical world is also reflected by methods for time synchronization and location estimation themselves. For example, methods for location estimation based on the measurement of time of travel or

time difference of arrival of certain signals typically require synchronized time. The other way round, location information may also help to achieve time synchronization. This is due to the fact that time synchronization approaches often have to estimate message delays. One component of the message delay is the time of flight of the carrier signal between two nodes, which can be calculated if the distance between sender and receiver and the propagation speed of the carrier signal are known.

4.1 Locating nodes in Space and Time

A common model for location estimation and time synchronization is presented here. Using this model, various requirements on and different classes of time synchronization and localization are discussed. One possible way to model physical space is to do this as a three-dimensional real-valued vector space. Physical time can be modeled as a one-dimensional real-valued vector space. These two vector spaces are often combined to form a four-dimensional vector space, known as *space-time.* To indicate points in space-time, a coordinate system is used, consisting of the vector o and four linearly independent vectors e_1, e_2, e_3, e_4 (the axes). For simplification, coordinate system has the following properties: $e_4 = (0, 0, 0, t)$, are mutually orthogonal and $|e_1| = |e_2| = |e_3|$.

The space axes e_1, e2, e3 form a Cartesian coordinate system, e_4 is the time axis, and $|e_1| = |e_2| = |e_3|$ and $|e_4|$ are the space and time units, respectively. Any point p in space-time can now be specified by its coordinates (p_1, p_2, p_3, p_4) with respect to the coordinate system (o, e_1, e_2, e_3, e_4), such that p is given by $o + p_1e_1 + p_2e_2 + p_3e_2 + p_4e_4$.

Under these assumptions, the spatial distance between two points p and q is given by:

$$\{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2\}^{1/2} \quad (1)$$

and the temporal distance is given by:

$$|p_4 - q_4|. \quad (2)$$

The above model allows a unified view on localization and time synchronization as follows. If a sensor node is modeled as a point p in space-time, localization and time synchronization can be considered as determining the current coordinates of p with respect to a given coordinate system. It is quite common to use different coordinate systems, even in a single application. However, using a simple coordinate transformation scheme, the coordinates p_i of a given point p can be transformed into coordinates p_i in a primed coordinate system, as it is shown on Figure *2,* for a two-dimensional coordinate system.

Localization in space-time comes in many different flavors and with many different requirements and practical constraints.

4.2 Distributed Algorithms for Time-Space Localization

There have been numerous algorithms proposed for *WSNs* that rely on localization data. In this section, we provide a brief survey of them; we divided them into four categories based on their functionalities: one-cast routing, multicast routing, energy consideration, and network security

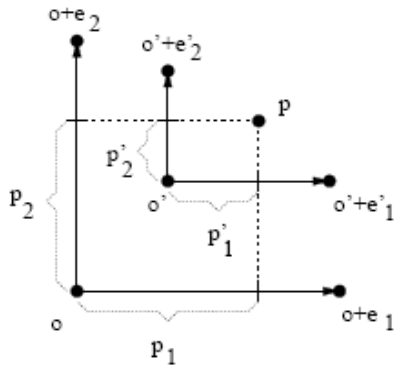


Fig. 2: A point p in space-time and its coordinates p_i and p_i' in two different coordinate systems.

.Figure 3 illustrates client nodes of the network with their location in time and space.

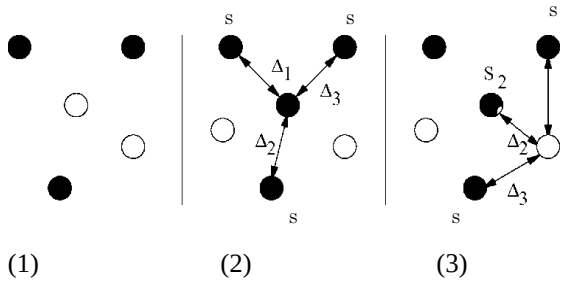


Fig. 3: Client nodes infer their location in space-time by measuring relationships in space-time

(1) presents two kinds of nodes: black *reference nodes* with known locations and white *client nodes* with unknown locations. In *(2),* a gray client node measures its distance A_j from a number of neighboring reference nodes. Using the locations S_j of the references and the measured distances *Aj*, the gray node infers its own location in space-time. The client node can now also act as a reference for other client nodes in subsequent iterations of the algorithm as illustrated in *(3).* All nodes should be able to measure distances to a sufficient number of neighboring reference nodes in order to estimate their location in space-time.

The meaning of the symbols *A* and *S* has to be interpreted in a rather broad sense here. *S* is any state information of a node that is relevant to a localization algorithm. Examples for *S* are time, location, orientation, and address of a node. *S* may also include confidence values that characterize the precision of the respective bits of state information. *A* is a space-time relationship between a client node and one or more reference nodes. Examples include Euclidean distance, hop distance, message delay, and angle with respect to the orientation of the client. *A* may also include confidence values.

A pair *(S, A)* can be interpreted as a *constraint* on the possible space-time locations of a client node. If *S* is a location of a

reference node in space and *A* its Euclidean distance, then the location of the client node is constrained to the hull of a sphere with radius A centered at *S*. A constraint may also involve multiple reference nodes, such that *A* is a relationship among a client node and any number of reference nodes. Also, reference nodes need not be network neighbors of the client node.

An important element of localization algorithms is a procedure for combining multiple constraints. As pointed out above, a single constraint limits the possible locations of a client node, but the resulting solution space often does not satisfy precision requirements. Multiple constraints have to be combined to further cut down the solution space.

An important component of localization algorithms also is rules determination for selecting constraints. In dense networks with many reference nodes, there is a large set of possibilities for obtaining constraints that involve different sets of reference nodes. While a large number of constraints may result in very precise location estimates, the overhead for combining such numerous constraints may be prohibitive. The goal is to select a small number of tight constraints that are sufficient to achieve a certain precision. Certain reference nodes may only become available after they have estimated their location themselves. An overlay structure is constructed to ease this selection process. a client node may use its parent in a spanning tree as a reference node. Essentially, constraint selection can be interpreted as the approach an algorithm takes to structure localization in multi-hop networks. Important element of localization algorithms is an approach to maintain localization over time, since a single estimate of a node's location in space-time is quickly invalidated due to the progress of time and due to node mobility. The conceptually simplest approach to this problem is to repeat a one-shot localization frequently. Bootstrapping mechanism is needed to provide initial reference nodes that act as seeds for distributed localization algorithms.

Many practical algorithms consist several phases in order to improve precision or other performance metrics. Several algorithms consist of a first phase to obtain rough location estimates for all nodes. In the next explained phase, the so-called refinement phase, these initial estimates are further improved.

4.3 Maintaining Localization over Time

A single run of a localization algorithm allows each node to estimate its location in space-time at a certain point in real time. However, as time progresses, the precision of this one-shot estimate may decrease quickly due to node mobility or due to the progress of time. Obviously, an algorithm can be executed one more time to obtain up-to-date estimates. The resulting precision over time then depends on the frequency of execution. However, since each execution of the algorithm takes a certain amount of time, this frequency cannot be arbitrarily increased. Hence, the maximum precision over time is also limited. Alternatively, if a certain target precision is requested by the application, the execution frequency may be calculated to be just high enough to provide the requested precision. For localization in space it is also possible to limit re-execution to nodes that have changed their location [3] in the meantime.

One way to further improve precision over time is the use of sensors to measure the location in space-time locally without referring to other nodes. This technique is also known as *dead reckoning.* Hardware clocks are dead-reckoning devices for estimating the current time. Accelerometers may be used to measure movements and can hence provide estimates of the current position in space [11]. Dead-reckoning techniques typically suffer from significant errors that accumulate over time and can therefore only be used to bridge the short gap between two consecutive runs of a localization algorithm. Typical hardware clocks suffer from an unknown clock drift between *10* and *100* parts per million. After one minute, the deviation from real time is then between *0.6* and *6* milliseconds. For location estimation using accelerometers, there is a quadratic relationship between acceleration-measurement errors and errors in the computed location estimate.

Another way of improving the precision is *prediction,* where based on location estimates from the past a current estimate is computed. Besides the past behavior, prediction requires a model of how a node can move through space-time. With respect to time, such a model is rather simple as real-time progresses at a constant rate. The situation gets more complicated for space, where nodes can move in complex patterns. It is often possible to derive constraints on the possible locations, bounds on speed and acceleration. If there is an upper bound on the speed of a node, we can derive bounds on the possible locations of a node at time t_i given the node's location at time $to < t_i$. Prediction can be achieved by fitting a curve to a set of locations in space-time observed in the recent past. As with dead reckoning techniques, prediction often experiences significant errors.

5. Time synchronization

The significance of physical time for sensor networks has been reflected by the development of a number of time synchronization algorithms in the recent past. Most of these approaches have been designed for "traditional" sensor networks. Most computer systems in use today are based on clocked circuits and hence contain so-called *digital clocks.* Such hardware clocks are a valuable tool for time synchronization, since they can be used to maintain *synchronization over time*.

A typical hardware clock consists of a quartz-stabilized oscillator and a counter that is incremented by one every oscillation period. If the periodic time *T* of the oscillator is known, the counter h can be used to obtain approximate measurements of real-time intervals in multiples of *T*.

The clock counter displays value *h(t)* at real time t and is incremented by one at a frequency of f. The rate of the counter is defined as $f(t) = dh(t)/dt$. An ideal digital clock would have a rate of 1 at all times. The periodic time of the oscillator and hence the clock rate depend on various parameters such as age of the quartz, supply voltage, environmental temperature and humidity.

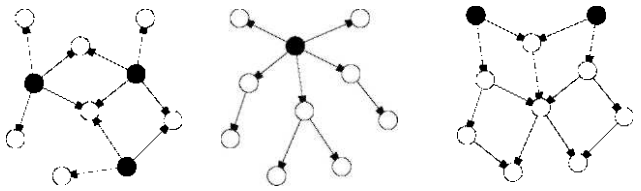
This so-called *clock drift* is formally defined as the deviation of the rate from *1* or:

$$\rho(t) = f(1) - 1 \qquad (3)$$

Since sensor nodes are typically operated under a well-defined range of the above parameters, it is reasonable to assume a *maximum possible drift* ρ_{max}, such that:

$$|\rho(t)| \leq \rho_{max} \quad (4)$$

Obtaining temporal constraints is typically implemented by communication among sensor nodes.
In practice, the relationship between synchronized time and hardware clock is often not linear. By repeating the line fitting procedure frequently, a linear approximation of that nonlinear relationship can be achieved. This approximation is the better, the fewer data points are included in each fitting procedure. Commonly used overlay topologies are shown on Figure *4*.



(a (b (c

Fig. 4. Commonly used overlay topologies. (a) Stars, (b) tree, (c) hierarchy

Often it is important to ensure that this approximation is continuous, which may require the introduction of additional constraints on the fitted lines.
The hardware clock can be considered a *time sensor*, calibrated using the observed past behavior of synchronized time.
The *precision* of the chosen approach should be valuated and the imprecision of time synchronization algorithm should be decreased, depending on the age of time marks and hop-distance between nodes of the sensor network, providing accuracy ordered in milliseconds.
One possible way of presenting *time synchronization* using programming language *C* is:

```
struct TimeMark {
Time begin, end, received; Time t1, t2, t3;
};
```

Begin and end are the left and right ends of the time-mark interval, received is the time of arrival, where t_1, t_2, t_3 are three sums incrementally calculated as the message is forwarded from node to node. Begin, end and received are local variables that don't need to be transmitted between nodes. Time is a representation for points in time and time differences. Computer clocks are discrete, so an integer type would be appropriate. The generator of a time-marked message performs the following actions:

```
TimeMark T;
T.begin = T.end = T.received = NOW;
T.t1 = T.t2 = t3 = 0;
```

NOW refers to the current value of the local clock. The interval is initialized to current time in the node. All other fields are set to zero. A message is sent using the following actions:

```
Sender:
        TimeMark T;
        Time idleend = NOW;
        IF (idlebegin[receiver] == 0 OR
            idleend - idlebegin[receiver] > max_idle)
        THEN
            send <sync> to receiver;
            receive <ack> from receiver;
              idleend = NOW;
              idlebegin[receiver] = idleend;
        ENDIF

        send <xmit(T, idleend - T.received,
                  idleend - idlebegin[receiver],
                  local_rho)> to receiver;
        receive <ack(resend)> from receiver;
      idlebegin[receiver] = NOW;

      IF (resend == TRUE) THEN
            idleend = NOW;
            send <xmit(S, idleend - T.received,
                        idleend - idlebegin[receiver],
                        local_rho)> to receiver;
            receive <ack> from receiver;
            idlebegin[receiver] = NOW;
      ENDIF
```

The sender first checks if the time when the last message was sent to the receiver is unknown or if the *idle time* is too large. Then the sender transmits data structure to the destination node along with the amount of time the message was stored in the current node and the idle time according to the local time scale with maximum clock drift *local_rho.* If resend is true, then the message is sent again in order to enable the receiver to measure round-trip time. The receiver of a message performs the actions:

```
Receiver:
      IF (receive <sync> from sender) THEN
            rttbegin[sender] = NOW;
            send <ack> to sender;
      ELSEIF (receive <xmit(T, lifetime, idletime,
                                rho)> from sender)
    THEN
            Time rttend = NOW;
            IF (rttbegin[sender] == 0) THEN
                  rttbegin[sender] = NOW;
                  send <ack(TRUE)> to sender;
    ELSE
                  T.t1 += lifetime/(1 - rho);
```

```
            T.t2 += idletime/(1 + rho);
            T.t3 += lifetime/(1 + rho);
            T.begin = rttend
                - T.t1*(1 + local_rho)
                + T.t2*(1 - local_rho)
                - (rttend - rttbegin[sender]) + D;
            T.end = rttend
                - T.t3*(1 - local_rho)
                - D;
            M.t1 += (rttend - rttbegin[sender] - D)
                * (1 - local_rho);
            M.t3 += D*(1 + local_rho);
            rttbegin[sender] = NOW;
            send <ack(FALSE)> to sender;
        ENDIF
    ENDIF
```

The receiver waits for a *sync* or *xmit* message from a sender. If it receives a sync, it just initializes *rttbegin[sender]* and returns an *ack* to the sender. *D* presents delay. The receiver waits for a sync or xmit message from a sender. If it receives a *sync,* it just initializes *rttbegin[sender]* and returns an *ack* to the sender.

If an *xmit* message is received, then the receiver first checks if the time of arrival of a previous message from this sender is known. If not so, then *rttbegin[sender]* is initialized and an *ack(TRUE*) message is returned to the sender, asking for a retransmission.

If rttbegin[sender] is known in the sender, then the fields of the received time mark *T* are updated and *T.begin* and *T.end* are calculated. Only after *T.begin* and *T.end* have been calculated, $M.t_1$ $(M.t_3)$ are updated accordingly. Finally, an ack is sent back to the sender.

6. *WSN* Topologies

Several wireless sensor networks topologies are considered in our work. The first network topology presented here is C – random network topology. Together with the presented H – random topology, they are irregular network topologies, as well as the random topology.

The C and H – random topologies are random topologies, but irregular once, because only a part of the square of the surface is considered.

Disturbed grid topology and disturbed hexagonal topology are more regular network topologies than previous mentioned ones.

7. Time-space localization algorithm with a mobile beacon approach

An original approach for time space localization, for different network topologies, is presented in this section. The positioning component is essentially the Mobile beacon localization algorithm [22]. The synchronization component obeys the same principle as the positioning algorithm, but it is extended to deal with time estimations. Then, these two components are combined into a single algorithm. The delay measurement technique is used to synchronize nodes.

Different network topologies are employed, generated by our algorithm.

In this algorithm, once nodes are deployed, the mobile beacon travels through the sensor field broadcasting messages that contain its current coordinates and timestamp. When an unknown node receives a packet from the mobile beacon, it can estimate the packet travel time and, based on the times tamp stored in the packet, its own offset. Also, when an unknown node receives more than three messages from the mobile beacon, it can estimate its position based on the received coordinates and on the *RSSI* distance estimates. Since the nodes will require at least three packets for positioning, synchronization can be improved by computing the average offset of all packets. First, set of position information is given as a variable, then set of received timestamps follows. Timer to send packets is put, also position and time information through variables is given. After that, the nodes' *GPS* info is returned, packets' travel distance estimation is given, the nodes' position is computed. Later, packets' travel delay estimation is given and also the nodes' offset is computed. Case if this node is a beacon node is explored and number of references is tested, if there are three received messages from mobile beacon, it confirms enough references and proceeds with the algorithm.

The options for choosing five types of sensor networks is put here whether the user wants to choose grid oriented network, randomly chosen network, hexagonal network topology, or H and C random network topologies, presented in the previous section. Also, information about the parameters used and display options are given and choice by the user, according to the parameters and network topology, is enabled.

The evaluation of our algorithm is done, by performing simulations..

We did an experiment within the sensor field *92 x 92* m^2. *256* nodes are employed, for different network topologies, we have chosen five types of network topology, explained previously, for this evaluation. The density of a sensor network is picked to be *0,03 nodes/*m^2, and the communication range is *15 m*.

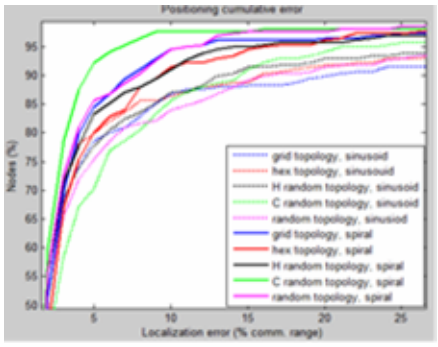
The evaluation is done by taking two types of trajectories for the mobile beacon: sinusoidal and spiral trajectories.

Localization error, synchronization error and also impact of *RSSI* inaccuracy are presented, in different color, for two evaluated trajectories of the beacon, sinusoidal an spiral ones, and for five types of network topologies: C – random topology, disturbed grid topology, disturbed hexagonal topology, H – random topology and random topology.

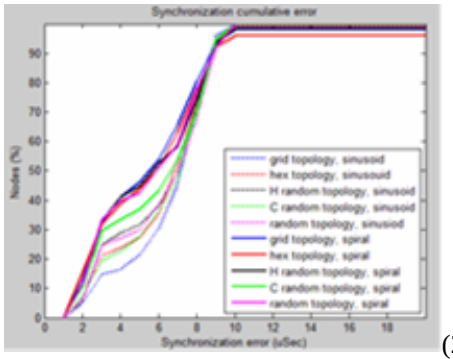
Localization Error - the distribution of position errors among the sensor nodes is depicted in Figure 5 *(1*). The cumulative error identifies the percentage of nodes (y-axis) with a positioning error. This is smaller than a parameterized value (*x*-axis). A sharp curve means that the majority of nodes has a small error. This graph also shows that spiral trajectories result in better positioning than sinusoidal ones, since these trajectories are less rectilinear.

Synchronization Error - the distribution of synchronization errors among the sensor nodes is depicted in Figure 5 *(2).* In this case, the cumulative error identifies the percentage of nodes (*y*-axis) with a synchronization error smaller than a parameterized value (x-axis). Again, a sharp curve means that the majority of nodes have a small error.

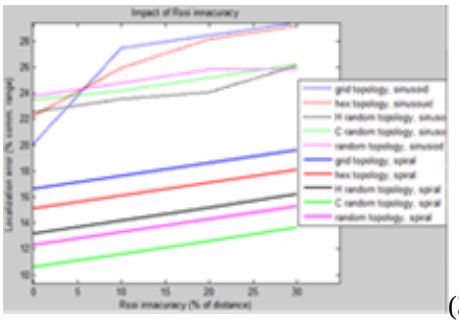
Impact of *RSSI* Inaccuracy – since the fact that the distance estimations using *RSSI* measurements are not accurate, depending on the environment, such an inaccuracy may lead to greater errors in the estimated positions. The evaluation of this impact is done



(1)



(2)



(3)

Fig. 5. *(1)* Positioning cumulative error *(2)* Synchronization cumulative error *(3)* Impact of *RSSI* inaccuracy

by adding some noise to the real distances. This noise is generated by a normal distribution, with the actual distance as the mean and a percentage of this distance as the standard deviation. The comparison between the increase of the standard deviation of the normal distribution (used to simulate the noise) and the actual distance for the algorithms is presented on Figure 5 *(3).* It can be noticed that the positioning part takes advantage of the synchronization part to improve its performance, which shows the significance of solving both positioning and synchronization problems at the same time.

8. Conclusions

Due to the close integration of sensor networks with the real world, the categories *time* and *location* are fundamental for many applications of sensor networks, to interpret sensing results or for coordination among sensor nodes. Time synchronization and sensor node localization are fundamental and closely related services in sensor networks.

Existing solutions for these two basic services have been based on a rather narrow notion of a sensor network as a large-scale, ad hoc, multi-hop, un-partitioned network of largely homogeneous, tiny, resource-constrained, mostly immobile sensor nodes that would be randomly deployed in the area of interest. However, recently developed prototypical applications indicate that this narrow definition does not cover a significant portion of the application domain of wireless sensor networks.

Existing solutions for time synchronization and node localization do not cover all parts of space and time in wireless sensor networks problem. Different, proposed approaches should be implemented to support these concepts adequately.

In this paper, we have proposed and evaluated an original algorithm for the time-space localization, a mobile beacon approach, for different network topologies. By using a mobile beacon, all sensor nodes are able to localize themselves both in time and space. The beacon node sends packets and all regular nodes are able to synchronize and compute their positions with a zero communication cost algorithm. The proposed algorithm shows the importance of combining both positioning and synchronization into a single unified problem: localization in time and space. Implementing this approach, communication and processing resources can be reduced, thus saving energy and network resources.

References

[1] M. Beigl, H.W. Gellersen, and A. Schmidt. MediaCups: Experience with Design and Use of Computer-Augmented Everyday Objects. *Computer Networks, Special Issue on Pervasive Computing,* 25(4):401-409, March 2001.

[2] P. Blum, L. Meier, and L. Thiele. Improved interval-based clock synchronization in sensor networks. In *Proceedings ofthe Third International Symposium on Information Processing in Sensor Networks (IPSN '04),* 2004.

[3]M. Bauer, L. Jendoubi, and O. Siemoneit. Smart Factory Mobile Computing in Production Environments. In *WAMES 2004,* Boston, USA, June 2004.

[4] C. Savarese, J. M. Rabaey, and K. Langendoen. Robust Positioning Algorithms for Distributed Ad-Hoc Wireless Sensor Networks. In *USENIX Annual Technical Conference,* Monterey, USA, June 2002.

[5] P. Basu and T. D. C. Little. Networked Parking Spaces: Architecture and Applications. In *VTC Fall,* Vancouver, Canada, September 2002.

[6]BerkeleyMotes.www.xbow.com/Products/Wireless_SensorJNetworks.htm.

[7] M. Hazas and A. Ward. A Novel Broadband Ultrasonic Location System. In *Ubicomp 2002,* Goteborg, Sweden, September 2002.

[8] CarTalk 2000. www.cartalk2000.net.

[9] L. Girod, V. Bychkovskiy, J. Elson, and D. Estrin. Locating Tiny Sensors in Time and Space: A Case Study. In *International Conference on Computer Design (ICCD) 2002,* Freiburg, Germany, September 2002.

[10] FleetNet. www.fleetnet.de.

[11] E. Vildjiounaite, E. J. Malm, J. Kaartinen, and P. Alahuhta. Location Estimation Indoors by Means of Small Computing Power Devices, Accelerome-ters, Magnetic Sensors, and Map Knowledge. In *PERVASIVE 2002,* Zurich, Switzerland, August 2002.

[12]K.Langendoen and N. Reijers. "Distributed Localization in Wireless Sensor Networks: a Quantitative Comparison," *Computer Networks,* vol 43(4), pp. 499-518, Nov. 2003.

[13] N. Bulusu, J. Heidemann, and D. Estrin. "GPS-less Low Cost Outdoor Localization for Very Small Devices," *IEEE Personal Communications Magazine,* vol. 7/5, pp. 28-34, October 2000.

[14] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher. "Range-Free Localization Schemes in Large Scale Sensor Networks," in Proc. of *ACM MOBI-COM'03,* pp. 81-95, 2003.

[15] D. Niculescu and B. Nath. "Ad Hoc Positioning System (APS)," in Proc. of *IEEE GLOBECOM'01,* vol. 5, pp. 2926-2931, San Antonio, 2001.

[16] D. Niculescu and B. Nath. "DV Based Positioning in Adhoc Networks," *Telecommunication Systems,* vol. 22, no. 1-4, pp. 267-280, January-April 2003.

[17] C. Savarese, J. Rabay, and K. Langendoen. "Robust Positioning Algorithms for Distributed Ad-Hoc Wireless Sensor Networks," in Proc. of *USENIX Technical Annual Conference,* pp. 317-327, Monterey, CA, June 2002.

[18] S. Capkun, M. Hamdi, and J.-P. Hubaux. "GPS-Free Positioning in Mobile Ad-Hoc Networks," in Proc. of *34th Hawaii International Conference on System Sciences,* vol. 9, pp. 9008, 2001.

[19] K. Chintalapudi, R. Govindan, G. Sukhatme, and A. Dhariwal. "Ad-Hoc Localization Using Ranging and Sectoring," in Proc. of *IEEE INFOCOM '04,* pp. 2662-2672, April 2004.

[20] A. A. Ahmed, H. Shi, and Y. Shang. "SHARP: A New Approach to Relative Localization in Wireless Sensor Networks," in Proc. of *25th IEEE International Conference on Distributed Computing Systems Workshops (ICDCS 2005),* pp. 892-898, June 2005.

[21] S. Thrun. "Particle Filters in Robotics," in Proc. of *18th Annual Conference on Uncertainty in Artificial Intelligence (UAI-02),* pp. 511, San Francisco, CA, August 2002.

[22] M. L. Sichitiu and V. Ramadurai. Localization of wireless sensor networks with a mobile beacon. In MASS’04, pages 174–183, Florida, USA, October 2004.