Determination of Time and Location in Large-Scale Dynamic Networks of Tiny Sensors

Kay Römer
Institute for Pervasive Computing
Department of Computer Science
ETH Zurich, Switzerland
`roemer@inf.ethz.ch`

Abstract. *So-called sensor nodes combine sensors, processors, wireless communication capabilities, and autonomous power supply in a tiny device. Large-scale networks of these untethered devices can be deployed unobtrusively in the physical environment in order to monitor a wide variety of real-world phenomena with unprecedented quality and scale. A fundamental service in sensor networks is the determination of time and location of events in the real world. This task is complicated by various challenging characteristics of sensor networks, such as their large scale, high network dynamics, restricted resources, and restricted energy. We develop new approaches for determination of time and location under these constraints, and devise design principles based on our experience. We illustrate the practical feasibility of our approaches by a concrete application.*

1. Introduction

So-called sensor nodes combine means for sensing environmental parameters, processors, wireless communication capabilities, and autonomous power supply in a single tiny device. Large-scale networks of these untethered devices can be deployed unobtrusively in the physical environment in order to monitor a wide variety of real-world phenomena with unprecedented quality and scale while only marginally disturbing the observed physical processes.

It is anticipated that a number of application domains can substantially benefit from the use of sensor networks. Biologists, for example, want to monitor the behavior of sensitive animals in their natural habitats; environmental research needs better means for monitoring environmental pollutions; agriculture can profit from better means for observing soil quality and other parameters that influence plant growth; geologists need better support for monitoring seismic activity and its influences on the structural integrity of buildings; and of course the military is interested in monitoring activities in unaccessible areas.

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, for example to identify physical events (i.e., infer time and location of occurrence of a physical event), to separate physical events (i.e., tell apart different physical events based on their distance in time and space), or to collaboratively monitor physical events (i.e., fuse sensory data from various sensor nodes). We will discuss these uses of time and location in Section 1.2..

On the other hand, sensor networks are subject to severe resource and energy limitations (due to the small size of sensor nodes), to network dynamics (due to node failures, node mobility, and environmental obstructions), and to scalability issues (due to the anticipated large number of sensor nodes participating in a network). We will discuss these issues in Section 1.1..

Our thesis is that *sensor networks require new approaches to determine time and location and it is actually possible to provide appropriate mechanisms in large-scale and highly dynamic sensor networks.* We support this thesis by describing what exactly makes determination of time and location different and difficult to achieve in the sensor network domain, by developing concrete algorithms, protocols, and prototype implementations for determination of time and location in sensor networks. The practical feasibility of our approaches is demonstrated by means of a typical sensor network application. We also formulate design principles that capture our experience with the development of mechanisms for the determination of time and location in sensor networks.

1.1. Sensor Network Challenges

Currently available sensor node prototypes are mainly built of commercially available components, resulting in matchbox-sized devices containing sensors, an embedded processor with few MIPS, few kilo bytes of RAM, and a radio transceiver with a communication range of some tens of meters and a shared bandwidth of some tens of kilobits per second. These devices run for weeks or few months on a set of batteries and cost about 100 Euros each [14].

However, these prototypical devices are often too large, too costly, or too short-lived for many anticipated applications of sensor networks. For example, applications may require sensor nodes that are light enough to stay suspended in air, to be small enough to be unobtrusively placed in the environment (e.g., mixed into paint and other coatings), to be cheap enough to deploy thousands of nodes, or to be energy-efficient enough to remain operational for years without changing batteries. Unfortunately, it is not easily possible to adopt the above mentioned prototypes to these requirements. First “Smart Dust” prototypes [12] implement sensor node functionality within few cubic millimeters and demonstrate that the tremendous volume reduction may require radical changes in the employed technologies (e.g., use of optical instead of radio communication).

Below we outline major challenges in the design of systems based on networks of such sensor nodes.

Restricted Size, Cost, Energy, and Resources. Application requirements often imply that sensor nodes be as small and as cheap as possible. This has a number of important implications. First of all, the amount of energy that can be stored in or harvested by devices with a small volume is very limited due to the low energy density of available and foreseeable technology. To ensure longevity despite this limited energy budget, energy-efficient design both in hardware and software becomes a dominating goal in sensor networks. Additionally, computing, storage, and communication capabilities of individual sensor nodes are very limited due to size and energy constraints. The use of common technologies such as GPS or even radios on sensor nodes may be precluded due to their prohibitively large size (e.g., of radio antennas), due to high cost (e.g., of GPS receivers), or due to high energy requirements (e.g., GPS receiver). Note that future technological advancements will likely be used to reduce size and energy consumption rather than to improve the capabilities of sensor nodes.

Network Dynamics. Due to the deployment in the physical environment, sensor networks are subject to a high degree of network dynamics. Sensor nodes can be mobile, sensor nodes die due to depleted batteries or due to environmental influences (e.g., stepping on the device; excessive pressure, humidity, heat; destructive chemicals), new sensor nodes are added to compensate for failed ones. This results in frequent changes in the network topology, in temporary network partitions, but also in long and unpredictable network latency. Dealing with a constantly changing network environment is thus another important design goal in sensor networks.

Scale of Deployments. Traditional remote sensing approaches used few high-resolution, long-range sensors to observe a large geographical area. The improved quality and detail of monitoring results obtained by sensor networks is mainly based on the fact that many "simple" sensors are located very close to the observed physical phenomenon. The small effective range of these sensors and a typically large geographical area of interest imply that sensor networks contain large numbers of densely deployed sensor nodes. Hence, scalability to large and dense networks is a further important design goal in sensor networks.

Unattended, Untethered Operation. In many applications, sensor networks have to be deployed in remote, unexploited, or hostile regions. Sensor networks therefore cannot rely on well-engineered or excessive hardware infrastructure (e.g., regular grids of beacons devices or base stations, wired power supply, wired communication). After initial deployment, it is often infeasible to physically access individual sensor nodes for hardware or software maintenance. The large number of nodes also precludes manual configuration of individual nodes (e.g., calibration of individual nodes). Ad hoc operation and self-configuration are therefore important design goals.

1.2. The Need for Determination of Time and Location

Due to the close integration of sensor networks with the real world, the real-world categories time and space play a crucial role for many sensor network applications. While many traditional uses of time and location also apply to sensor networks, we will focus here on areas of particular importance for sensor networks. These applications require common reference systems for time and location among sensor nodes.

Data Evaluation. For many applications, certain well-defined state changes in the real world – so-called physical events – are of interest (e.g., some object of interest appears or disappears). In order to *identify* the real-world cause of an event reported by the sensor network, it is often crucial to know time and location of occurrence of the reported event. Sensor networks should also be able to *separate* distinct physical events. For this, it must be possible to decide whether or not two sensor readings belong to the same physical event. This decision process is often based on time and location of a sensor node at sensor readout. Additionally, the observation of many phenomena in the physical world requires that sensory data from various nodes is *fused*. The velocity of a moving object can, for example, be estimated by correlating time and location of two or more object sightings by different sensor nodes. As this example illustrates, time and location can be fundamental for the fusion of sensory data.

Addressing. Since sensor networks are embedded into the real world, it is often convenient to address parts of the network by characteristics of the real world such as time and location, rather than using unique identifiers (e.g., "www.sensor500.net") as in traditional distributed systems. As one example, applications are often only interested in physical events occurring in certain regions in space and time (e.g., only during the night in a certain geographical area; a geographical area which is a function of time and vice versa). Nodes which are currently not contained in such a region of interest can be switched into a power-saving idle mode. Hence, time and location are important tools to assign tasks to sensor networks in energy-efficient ways.

Distributed Coordination. Collaboration among sensor nodes may require to trigger actions on a set of sensor nodes in a coordinated way. Consider for example the power-efficient use of wireless radios.

Since radios are one of the most power-consuming components of sensor nodes even in idle mode, a convenient solution is to regularly switch off the radio completely. However, radios must be switched on and off in a coordinated way, such that communication is still possible. Common solutions such as duty-cycling the radio or selectively switching off nodes in areas of high node density require a common understanding of time and location among sensor nodes. Distributed coordination may also be required for the control of sensors (e.g., on/off, alignment, sampling rate) or actuators.

2. Time Determination

The Problem. Energy, size, and cost constraints typically preclude equipping sensor nodes with receivers for time infrastructure such as GPS [4]. Also, logical time [5] is not sufficient, since it only captures causal relationships between “in system” events, defined by message exchanges between event-generating processes. In contrast, phenomena sensed by sensor nodes are triggered by external physical events which are not defined by in-system message exchanges; physical time must be used to relate events in the physical world.

Time synchronization services for traditional distributed systems like NTP [6] are typically based upon a manually configured hierarchy of network nodes. At the top of the hierarchy are one or more so-called master nodes – canonical sources of time which are synchronized to each other via some out-of-band mechanism such as GPS. Nodes further down in the hierarchy are synchronized to this global time scale by evaluating “time beacons” received from their immediate parent(s). Such beacon messages are frequently sent by a network node to its child nodes, containing the current time of the parent at the time of message generation.

There are various problems with such an approach in the context of sensor networks. As noted above, equipping master nodes with infrastructure such as GPS receivers is typically not an option. In the case of one master (where no external infrastructure for out-of-band synchronization is required), synchronization paths tend to be very long due to the expected scale of sensor networks. This may lead to poor synchronization of nodes far away from the master node. Even worse, nodes which are close to each other, but are far away from the synchronization master, may experience a large synchronization error with respect to each other due to using different synchronization paths to the master with different synchronization quality. This can be a major problem, since co-located nodes tend to require accurate synchronization in order to correlate local sensor events.

Moreover, synchronization schemes like NTP are not optimized for energy efficiency. For example, the CPU is used continuously to perform frequency disciplining of the oscillator by adding small increments to the system clock. In addition, synchronization beacons are frequently exchanged, which also requires constantly “listening” to the network for such beacons. However, with low-power radios used in sensor networks, listening to, sending to, and receiving from the network all require significant amounts of energy. Also, the CPU may not be available if the processor is powered down to save energy.

The manually and statically configured synchronization topology used by NTP is not compatible with the network dynamics in sensor networks. The frequently changing network topology precludes static configuration, the unattended operation of sensor networks precludes manual configuration of individual nodes. Moreover, sensor networks are likely to be temporary partitioned due to node failures or environmental obstructions. Clocks in different partitions are poorly synchronized, which may lead to difficulties when trying to temporally correlate sensor events originating from different partitions after a rejoin of the partitions. Also, accurate estimation of message delay – a base function for many synchronization approaches – is complicated by highly variable and indeterministic network delays.

On-Demand Timestamp Synchronization. In contrast to most traditional approaches, with our approach sensor node clocks run unsynchronized, defining a local time scale each. Reading this clock results in a timestamp, a data structure that represents present time on the local time scale. Note that any point in time in the past and the future can be expressed by a timestamp (i.e., present time) and an offset relative to this timestamp. When a node sends a timestamp to a neighbor as part of a network message, the timestamp is transformed to the local time scale of the receiver in order to establish a common understanding of time among sender and receiver. By extending this approach to multiple hops, timestamps received from any node in the network can be reconciled with the local time scale of the receiver. This way, a sensor node establishes (on demand) a common time scale among timestamps received from nodes throughout the network.

Timestamp transformation is achieved by determining the age of each timestamp from its generation to arrival at a sensor node. On a multi-hop path, the age is updated at each hop. The timestamp can then be transformed to the receivers local time scale by subtracting the age from time of arrival.

The age of a timestamp consists of two components: (1) the amount of time the timestamp resides in nodes on the path, and (2) the amount of time needed to transfer the timestamp from hop to hop. The first component is easy to measure since only a single clock is involved. The second component can be bounded by the round trip time of the message and its acknowledgment. This can be often be achieved without additional message exchanges by piggybacking on existing messages and acks. By taking into account the drift of the local clocks, we can determine upper and lower bounds of the age. Hence, the transformation of a timestamp results in upper and lower bounds for the timestamp on the local time scale of the receiver.

This approach has several advantages over traditional clock synchronization schemes. It is energy-efficient, since it is demand-driven and piggybacks on existing message exchanges. It is scalable, since there is only local interaction among network neighbors. The algorithm can deal with network dynamics, since it does not rely on global structures or topologies that could be destroyed. It even works in the presence of temporary network partitions.

Due to the limited space, we can only sketch the basic ideas here. Details, implementation, evaluation, and related work of this approach are described in [7].

3. Location Determination

The Problem. As with time synchronization, energy, size, and cost constraints typically preclude equipping sensor nodes with receivers for location infrastructures such as GPS. In extreme cases such as Smart Dust [12], it might not even be possible to equip sensor nodes with transceivers for radio waves or ultra sound due to the tiny size and energy budget of Smart Dust nodes. For example, a radio antenna would be many times larger than a complete Smart Dust node. Hence, traditional ranging approaches such as ones based on time of flight of ultrasound signals or received radio signal strength might render unusable in the context of sensor networks.

Many location systems such as [11] depend on an extensive hardware infrastructure. Location systems based on trilateration, for example, require many spatially distributed and well-placed infrastructure components in order to achieve high accuracy, which is not an adequate solution for ad hoc sensor networks. Location approaches that require centralized computation or other central software components such as [2] do not scale well to large networks.

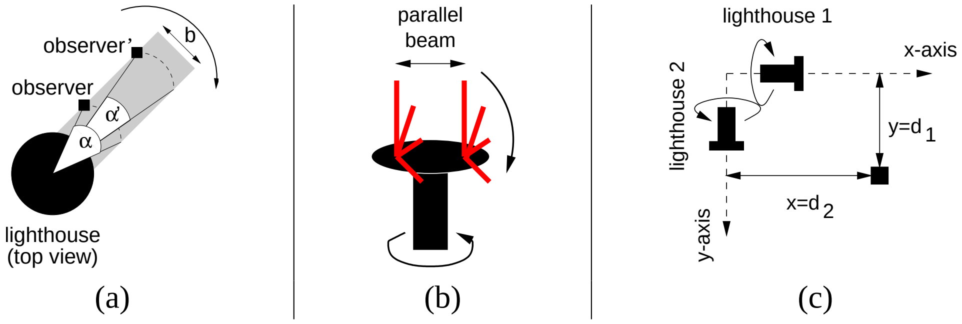

Figure 1. (a) Lighthouse with parallel beam (b) Lighthouse implementation with two rotating laser beams (c) 2D location system using two lighthouses.

To overcome the limitations of infrastructure-based approaches, various schemes for ad hoc location determination have been devised (e.g., [10]). They are typically based on the assumption that few nodes of the network – so-called anchor nodes – know their exact location via some out-of-band mechanism such as GPS. Other nodes derive their location by, for example, multilateration based on the distances to three or more neighbors with known locations. By iterating this process, all nodes of the network should eventually end up with three or more neighbors with known locations in order to be able to estimate their own location. To avoid accumulating errors inherent to such iterative approaches, many schemes calculate initial location estimates in a first round and iteratively improve these estimates in a number of additional rounds. However, there are several problems with such an approach. Firstly, good location estimates are only obtained if each node has many neighbors, i.e., if the network is dense. But even then, nodes at the edges of the network tend to end up with poor estimates since they have fewer neighbors. Secondly, the iterative nature of many of the algorithms typically implies a high message overhead, leading to poor energy efficiency. Thirdly, the iterative nature of these approaches and high network latency imply significant convergence times, which may lead to problems in dynamic and mobile networks.

An important overhead involved in setting up a location system is node calibration in order to enforce a correct mapping of sensor readings to location estimates [13]. In systems based on radio signal strength, for example, the received signal strength is mapped to a range estimate. Variations in transmit power and frequency among the nodes can cause significant inaccuracies in the range estimates when used without calibration. Since the cheap low-power hardware used in sensor nodes typically introduces a high variability between nodes, sensor nodes have to be individually calibrated. This, however, may not be feasible in large sensor networks.

The Lighthouse Location System. Our approach is specifically tailored to Smart Dust – tiny sensor nodes using optical communication with a volume of only few cubic millimeters. The basic element of the location system is a device similar to a lighthouse, but which emits a parallel beam (i.e., a beam with constant width) as depicted in Figure 1 (a). The sweep time (i.e., amount of time during which an observer sees the lighthouse flash) then is a function of the observer's distance from the lighthouse rotation axis. Knowing the width of the beam and the lighthouse rotation speed, we can calculate the distance of an observer (i.e., sensor node) from the lighthouse rotation axis.

A parallel beam can be implemented as depicted in Figure 1 (b): two rotating laser beams (at high speeds) define the outline of a wide "virtual" parallel beam, which in turn is rotating around a central axis (at much lower speeds) to create a rotating lighthouse effect. An observer looking at such a lighthouse sees two sequences of short laser flashes as the two "laser light planes" rotate by.

Using two such lighthouses, a 2D location system can be constructed as depicted in Figure 1 (c). The two lighthouses are assembled such that their rotation axes are mutually perpendicular. The distances d_1 and d_2 to the lighthouse rotation axes then equal the y and x coordinates of the observer in the 2-dimensional coordinate system defined by the lighthouse rotation axes. Accordingly, a 3D location system can be built out of 3 lighthouses with mutually perpendicular rotation axes.

This approach has a number of advantages. Only a single additional device is needed. Sensor nodes consume only few energy, since they do not actively emit any signals. Smart Dust prototypes demonstrate that an optical receiver requires few energy and size. The system is scalable and insensible to network dynamics, since sensor nodes autonomously determine their location without interacting with other sensor nodes. Also, sensor nodes need not be calibrated, since constant delays in the signal path cancel out due to differential measurements (i.e., we are only interested in the sweep time, which is the difference between two measured points in time).

Due to the limited space, we can only sketch the basic ideas here. Details, implementation, evaluation, and related work of this approach are described in [8].

4. Application Experience

In order to demonstrate the practical feasibility and to gain experience with our approaches for determination of time and location, we developed a prototypical application to track the movements of a mobile object with a sensor network, using a remote-controlled toy car as a sample target. Note that tracking is a commonly required function in many sensor network applications (e.g., tracking animals in their habitats, tracking an oil stain on the ocean, tracking a cloud of toxic gas).

Our application is tailored to tiny sensor nodes known as “Smart Dust” [12]. However, since Smart Dust hardware is not yet available, we used BTnodes [1] for the implementation of our prototype system. A number of these sensor nodes are randomly deployed in the area of interest. When they detect the presence of the car using attached sensors, they send notifications to a base station (a laptop computer). The base station fuses these notifications in order to estimate the current location of the car. A graphical user interface displays the track and allows to control various aspects of the system. The data fusion process requires that nodes share a common reference system in time and space. We apply the mechanisms described in the previous sections to establish such reference systems. Details, implementation, and evaluation of this approach are described in [9].

5. Design Principles

Based on our work on the design of approaches for determination of time and location in sensor networks, we tried to capture our experience in a number of general design principles. These could be helpful for future research in this domain. Due to limited space, we can only sketch these here, details can be found in [3].

Local Interaction. Avoiding or limiting interaction among sensor nodes to the immediate network neighborhood is a key for dealing with network dynamics, since network dynamics likely disturb non-local interaction. Timestamp synchronization, for example, only “knows” about immediate network neighbors and does not try to establish global time scales. The lighthouse location system avoids interaction among nodes.

Exploitation of Application Characteristics. Exploiting characteristics of the application is a key to resource-efficient and energy-efficient systems by providing services only where, when, and with the quality actually required by the application. This is not only true for static characteristics that don't change significantly over the lifetime of the application, but also for more dynamic characteristics. Timestamp synchronization, for example, only synchronizes where and when required by the application.

Exploitation of Platform Characteristics. Another key to resource efficiency and energy efficiency is to exploit characteristics of the hardware environment instead of designing one-size-fits-all solutions. The lighthouse location system, for example, is particularly tailored to Smart Dust, which does already contain the necessary hardware components such as an optical receiver.

6. Summary

We identified determination of time and location in large-scale dynamic sensor networks as an important research topic. We developed solutions for special instances of these problems and demonstrated their feasibility by means of a prototype sensor network application for tracking the location of mobile targets. We captured our experience in a number of design principles.

References

[1] J. Beutel, O. Kasten, F. Mattern, K. Römer, L. Thiele, and F. Siegemund. Prototyping Sensor Network Applications with BTnodes. In *EWSN 2004*, Berlin, Germany, January 2004.

[2] L. Doherty, K. S. J. Pister, and L. E. Ghaoui. Convex Position Estimation in Wireless Sensor Networks. In *Infocom 2001*, Anchorage, Alaska, April 2001.

[3] J. Elson and K. Römer. Wireless Sensor Networks: A New Regime for Time Synchronization. In *Hotnets-I*, Princeton, USA, October 2002.

[4] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning System: Theory and Practice, 4th Edition*. Springer-Verlag, 1997.

[5] L. Lamport. Time, Clocks, and the Ordering of Events in a Distributed System. *Communications of the ACM*, 21(4):558–565, July 1978.

[6] D. L. Mills. Improved algorithms for synchronizing computer network clocks. In *Conference on Communication Architectures (ACM SIGCOMM'94)*, London, UK, August 1994. ACM.

[7] K. Römer. Time Synchronization in Ad Hoc Networks. In *MobiHoc 2001*, Long Beach, USA, October 2001.

[8] K. Römer. The Lighthouse Location System for Smart Dust. In *MobiSys 2003*, San Franscisco, USA, May 2003.

[9] K. Römer. Tracking Real-World Phenomena with Smart Dust. In *EWSN 2004*, Berlin, Germany, January 2004.

[10] A. Savvides, C. C. Han, and M. Srivastava. Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors. In *Mobicom 2001*, Rome, Italy, July 2001.

[11] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The Active Badge Location System. *ACM Transactions on Information Systems*, 10(1):91–102, 1992.

[12] B. Warneke, M. Last, B. Leibowitz, and K. S. J. Pister. Smart Dust: Communicating with a Cubic-Millimeter Computer. *IEEE Computer Magazine*, 34(1):44–51, January 2001.

[13] K. Whitehouse and D. Culler. Calibration as Parameter Estimation in Sensor Networks. In *Workshop on Wireless Sensor Networks and Applications (WSNA) 02*, Atlanta, USA, September 2002.

[14] Berkeley Motes. www.xbow.com/Products/Wireless_Sensor_Networks.htm.