Chapter 1

SENSOR NETWORKS: A BRIDGE TO THE PHYSICAL WORLD

Jeremy Elson and Deborah Estrin
Center for Embedded Networked Sensing
University of California, Los Angeles
Los Angeles, CA 90095
{jelson,destrin}@cs.ucla.edu

1. The Quake

It was in the early afternoon of an otherwise unremarkable Thursday that the Great Quake of 2053 hit Southern California.

The earth began to rupture several miles under the surface of an uninhabited part of the Mohave desert. Decades of pent-up energy was violently released, sending huge shear waves speeding toward greater Los Angeles. Home to some 38 million people, the potential for epic disaster might be only seconds away. The quake was enormous, even by California standards, as its magnitude surpassed 8 on the Richter scale.

Residents had long ago understood such an event was possible. This area was well known for its seismic activity, and had been heavily instrumented by scientists for more than a century. The earliest data collection had been primitive, of course. In the 1960's, seismometers were isolated devices, each simply recording observations to tape for months at a time. Once or twice a year, seismologists of that era would spend weeks traveling to each site, collecting the full tapes and replacing them with fresh blanks. If they were lucky, each tape would contain data from the entire six months since their last visit. Sometimes, they would instead discover only a few hours of data had been recorded before the device had malfunctioned. But, despite the process being so impractical, the data gathered were invaluable—revealing more about the Earth's internal structure than had ever been known before.

By the turn of the century, the situation had improved considerably. Many seismometers were connected to the Internet and could deliver a continuous stream of data to scientists, nearly in real-time. Experts

could analyze earthquakes soon after they occurred, rather than many months later. Unfortunately, instrumenting undeveloped areas this way remained a problem, as networked seismometers could only be deployed where infrastructure was available to provide power and communication.

Some researchers at that time worked on early-warning systems that would sound an alarm in a city moments before an earthquake's arrival. If a sensor was close enough to the epicenter, and the epicenter was far enough from a population center, the alarm could be raised 20 or 30 seconds before the city started to shake. The idea was promising. But the lack of a pervasive sensor-support infrastructure was a serious impediment to deploying large-scale networks in an area like the Mohave desert.

But, in the half-century leading up to the Great Quake of 2053, technological advances changed everything. Pervasive infrastructure was no longer required for pervasive sensing. And, perhaps equally important, the job of responding to an alarm was no longer solely the domain of *people*. With the help of its new sensory skin, the city itself could protect its inhabitants.

By the mid 2040's, the vast, desolate expanse of the desert floor was home to nearly a million tiny, self-contained sensors. Each had a processor, memory, and radio that could be used to communicate and cooperate with others that were nearby. Most were almost invisibly small. There were dozens of varieties, each with a different observational specialty. In the desert, the atmospheric, chemical, organic, and seismic sensors were most popular.

It was just a few dozen of those seismometers—closest to the epicenter—that first sensed unusual acceleration in the ground, nearly the instant that shock from the Great Quake reached the desert's surface. Individual sensors could not be trusted, of course, but as the number of confirmed observations grew, so did the likelihood that this event was not simply random noise, or a malfunction. It was real. But what was it?

In those first few milliseconds, information was sketchy. To conserve energy, many sensors had been off. Those that happened to be active were only monitoring the ground with low fidelity. The network did not yet have enough detailed data to distinguish an earthquake from the demolition of a far-away building, or a boulder rolling down a hill.

More data were needed quickly, from a much larger area. Electronic word of the anomaly spread. In a few tenths of a second, the earth's movement had the full attention of thousands of seismometers within a few miles of the epicenter. In seconds, most saw the same shock waves as they raced past with a frightening magnitude. The network soon reached consensus: this was an earthquake. It was a dangerous one.

Small earthquakes are commonplace, and typically only of academic interest. For efficiency's sake, information of such quakes might not be transmitted outside of the desert for a minute or two. But a quake as big as that one in 2053 was a matter of public safety and time was of the essence. Seismometers immediately recruited the neighboring sensors that had the fastest, longest-range, highest-power radios. Several summarized the important details and sent urgent messages on their way. The information hopped from one node to the next, streaking wirelessly across the desert floor. After 41 miles, it finally reached the first sign of civilization: a wired communication access point. Four seconds had passed since the quake began. Once on the wired grid, the alarm spread almost instantly to every city in the area.

The new generation of smart structures in Los Angeles learned of the quake nearly thirty seconds before it arrived. Thousands of high-rise buildings, bridges, freeways, underground pipelines, and even some private homes were informed of the important details, such as the impending shock's magnitude and frequency composition. The alarm reached a dizzying array of embedded computers, from millions of microactuators attached to internal beams to the enormous adjustable shock-absorbers in building foundations. Structures throughout the city quickly de-tuned themselves so as to prevent resonance and collapse. These electronic earthquake countermeasures had been mandated by building codes for more than twenty years.

Tense moments passed. Sirens blared as every traffic light turned red and every elevator stopped and opened at the nearest floor. The city seemed to stand still, holding its breath. Finally, the silence was broken, at first by a low rumble, then a deafening roar. The earth rolled and shook violently. Sensors on the ground and within each structure monitored the dynamics of the motion. Each continued to make defensive adjustments. Swaying and groaning, buildings and bridges were strained, but most survived, keeping their occupants unharmed.

Even with the countermeasures, the city could not completely escape damage. Older buildings were particularly susceptible, and many older homes collapsed. Rescue crews arrived quickly with Portable Emergency Survivor Locators. Each was a nylon package the size of a wine bottle, containing thousands of tiny, self-propelled sensors that could disperse themselves as the package was thrown over or inside of the target area. Sensors were automatically activated when landing, and began to cooperatively explore their environment. Back at the rescue truck, a map of the structure began to appear. The structure itself was mapped using radar and acoustic reflections. People were visible as heat sources.

As sensors penetrated deeper into the the structure, the map grew and gained additional detail.

Sensors throughout the city's water system went onto high alert, ready to divert contaminants to a safe disposal area if any toxins were detected above the threshold safe for human exposure. An hour after the quake, chemical sensors in several older residential areas began to detect abnormal traces of natural gas—the result of ruptures in the labyrinth of pipes that snaked beneath the city. The newest pipes had sensors embedded every few inches along their outer surface, allowing instant and unambiguous location of leaks. But, many of the older pipes had not yet been replaced, so the far coarser job of detecting leaks with in-ground sensors began. After several minutes, the arrays were able to compute three-dimensional concentration gradients. They hypothesized the most likely set of points where pipes were broken. The upstream valves nearest to each point were commanded to close. On one city block, the shutdown did not come in time; the leak contaminated the area, risking an explosion. The sensor array warned residents and dispatched a crew to clean up the spill.

Meanwhile, in a forgotten corner of Los Angeles, a small fire started in an abandoned lumber yard. In another era, under the cover of the chaos, the flame might have gone unnoticed for hours, blazing wildly out of control. This particular fire, however, could not escape attention. Some years before, local high school students had scattered sensors that measured temperature and airborne particulates. They'd gathered data for a one-week class project on air pollution, but even now, unmaintained and long forgotten, some sensors still functioned. The combination of smoke and heat provoked their fire-warning reflex. A few miles down the road, in a sleepy volunteer fire department, a map to the yard popped up on a screen.

By Monday, Southern California had returned to normal. The 2053 quake came and went, thanks largely due to the pervasive sensors that had been woven into both our technological fabric and the natural environment. Even 50 years earlier, a similar quake would have caused widespread destruction. To many people in 2003, using technology to prevent such a catastrophe must have seemed fanciful and improbable. It seemed as improbable as 2003's globally interconnected and instantly searchable network of nearly a billion commodity computers must have seemed to those in 1953. Indeed, perhaps even as improbable as 1953's "electronic brain" must have seemed to those in 1903, who would soon experience their own Great Earthquake in San Francisco.

2. Observation of the physical world

Automatic detection, prevention, and recovery from urban disasters is but one of many potential uses for an emerging technology: *wireless sensor networks*. Sensor networks have captured the attention and imagination of many researchers, encompassing a broad spectrum of ideas. Despite their variety, all sensor networks have certain fundamental features in common. Perhaps most essential is that they are embedded in the real world. Sensors *detect* the world's physical nature, such as light intensity, temperature, sound, or proximity to objects. Similarly, actuators *affect* the world in some way, such as toggling a switch, making a noise, or exerting a force. Such a close relationship with the physical world is a dramatic contrast to much of traditional computing, which often exists in a virtual world. Virtual-world computers deal exclusively in the currency of information invented by humans, such as e-mail, bank balances, books and digital music.

Sensor networks are also large collections of nodes. Individually, each node is autonomous and has short range; collectively, they are cooperative and effective over a large area. A system composed of many short-range sensors lends itself to a very different set of applications than one that uses a small number of powerful, long-range sensors. The difference can be illustrated by considering an age-old question: is the sky clear or cloudy today? Answering this question is easy, even on a large scale, by using only a few long-range sensors such as satellite imagers. A satellite works because clouds have two important properties: they can be seen from far away, and a strategically placed sensor has an unobstructed view of a large area. A single satellite can detect the cloud cover of an entire hemisphere.

Such a centralized, long-range approach fail in complex, cluttered environments where line-of-sight paths are typically very short. For example, satellites are not very good at detecting individual animals in a forest, objects in a building, or chemicals in the soil. Moving to a distributed collection of shorter-range sensors can dramatically reduce the effect of clutter. By increasing the number of vantage points, it is more likely that an area will be viewable, even when line-of-sight paths are short.

Many interesting phenomena can not be effectively sensed from a long distance. For example, temperature and humidity are both very localized. Unlike clouds, they are not easily observable from afar, even by a sensor with a line of sight. Distributed sensing improves the signal-to-noise ratio because sensors are closer to the phenomena they are ob-

serving. This allows greater fidelity, and in the extreme, can reveal phenomena that are invisible when viewed from far away.

Distributed sensing has been successful for weather observations. For example, in the United States, temperature and humidity information is provided by over 600 automatic weather observation systems (AWOS) installed near airports. This distributed (but wired) sensor network produces local observations using short-range sensors. While the system is temporally continuous, it is spatially sparse, and therefore can not be used to sense phenomena at a greater resolution than several miles. This is an important and fundamental limitation: each sensor site requires a support infrastructure, including connections to the power and communications grids. The high overhead cost makes spatially dense sensing prohibitively impractical.

Wireless sensor networks can address the issue of observing the environment at close range, densely in space, and frequently in time. This has the potential to *reveal previously unobservable phenomena in the physical world*, making sensor networks attractive to a broad spectrum of scientists fAR03. Historically, new tools for observing the world around and within us have heralded new eras of understanding in science, as Galileo's telescope did for astronomy, or the microscope did for biology.

3. Technological Trends

Automated sensing, embedded computing, and wireless networking are not new ideas. However, it has only been in the past few years that computation, communication, and sensing have matured sufficiently to enable their integration, inexpensively, at low power, and at large scale.

Microprocessors have undergone a slow but inexorable transformation in the decades since their introduction. In 1965, Gordon Moore famously predicted that there would be an exponential growth in the number of transistors per integrated circuit. His rule of thumb still remains true. Year after year, manufacturers vie for customers' attention with products that boast the highest speed, biggest memory, or most features.

While such high-end technology has taken center stage, a dramatic but much quieter revolution has also taken place at the low end. Today's smallest processors can provide the same computational power as high-end systems from two or three decades ago, but at a tiny fraction of the cost, size and power.

In 1969, a system called the Apollo Guidance Computer was on board the manned spacecraft that made the journey from the Earth to the Moon's surface. The AGC was custom-designed over the course of ten years, and for some time afterwards was likely the most advanced com-

puter in existence. Its core had a 2 megahertz clock and could access 74,000 bytes of memory. Today, in 2003, a commercial off-the-shelf microcontroller with the same capabilities costs about $1, uses one ten-thousandth the power, and one hundred-thousandth the volume and mass.

That AGC-equivalent computational power has become so readily available is only part of the story. Small microcontrollers have been commonplace for so-called "embedded" applications for many years—computers, for example, that run digital radios, engine valve timing, and thermostats. Indeed, some 97% of processors manufactured are destined for such applications, rather than for personal computers on desks. But embedded processors have typically lived in isolation, each working independently. Without communication, their sole, tenuous link to one another is the humans who use them.

Wireless communication facilitates the integration of individual processors into an interconnected collective. The mass consumption of wireless devices has driven the technology through an optimization similar to microprocessors, with modern radios consuming less power, space, and money than their predecessors.

Sensor networks differ significantly in their communication model from typical consumer wireless devices. The primary focus of communication is nodes' interaction with *each other*—not delivery of data to the user. In sensor networks, a user does not micro-manage the flow of information, in contrast to browsing the Internet or dialing a cellular phone. Users are not aware of every datum and computation, instead being informed only of the highest-level conclusions or results.

This matches our experience in the natural world. For example, a person can feel a mosquito bite without being conscious of the individual sensations—a characteristic buzz, a flying insect seen peripherally, the tingling of hairs on the skin—that were synthesized to create the illusion of a mosquito bite "sensor." Similarly, in sensor networks, the goal is not necessarily to provide a complete record of every sensor reading as raw *data*, but rather to perform synthesis that provides high-level *information*.

The goal of delivering synthesized, high-level sensing results is not simply convenience. It is, in fact, a first-order design principle, because such designs conserve the network's most precious resource: energy.

4. Core Challenges

In sensor networks, energy is valuable because it is scarce. Deployment of nodes without a support infrastructure requires that most of

them be *untethered*, having only finite energy reserves from a battery. Unlike laptops or other handheld devices that enjoy constant attention and maintenance by humans, the scale of a sensor network will make manual energy replenishment impossible. Though certain types of energy harvesting are conceivable, *energy efficiency* will be a key goal for the foreseeable future. This requirement pervades all aspects of the system's design, and drives most of the other requirements.

Fundamental physical limits dictate that, as electronics become ever more efficient, *communication* will dominate a node's energy consumption PK00. The disproportionate energy cost of long-range vs. short-range transmission (r^2 to r^4), as well as the need for spatial frequency reuse, precludes communication beyond a short distance. The use of local processing, hierarchical collaboration, and domain knowledge to convert raw data into increasingly distilled and high-level representations—or, *data reduction*—is key to the energy efficiency of the system. In general, a perfect system will reduce as much data as possible as early as possible, rather than incur the energy expense of transmitting raw sensor values further along the path to the user.

Another fundamental challenge in sensor networks is their *dynamics*. Over time, nodes will fail—they may run out of energy, overheat in the sun, be carried away by wind, crash due to software bugs, or be eaten by a wild boar. Even in fixed positions, quality of RF communication links (and, thus, nodes' topologies) can change dramatically due to the vagaries of RF propagation. These changes are a result of propagation's strong environmental dependence, and are difficult to predict in advance. Traditional large-scale networks such as the Internet work in the face of changing configurations and brittle software partly because the number of people maintaining and using the network has grown along with the size of the network itself. In contrast, there may be a single human responsible for thousands of nodes in a single sensor network. Any design in which each device requires individual attention is infeasible. This leads to another important requirement: sensor networks must be *self-configuring*, working without fine-grained control from users. They must also be *adaptive* to changes in their environment—not make a single configuration choice, but continually change in response to dynamics.

5. Research Directions

These basic challenges in wireless sensor networks have sparked a number of different research themes. In this section, we give a broad overview of many of these areas. It is meant as a general guide to the types of work currently underway, not a comprehensive review.

Tiered architectures

Although Moore's law predicts that hardware for sensor networks will inexorably become smaller, cheaper, and more powerful, technological advances will never prevent the need to make tradeoffs. Even as our notions of metrics such as "fast" and "small" evolve, there will always be compromises: nodes will need to be faster *or* more energy-efficient, smaller *or* more capable, cheaper *or* more durable.

The choice of any *single* hardware platform will make compromises. The diverse needs of a sensor network can be satisfied only through a *tiered architecture*—a design that is a composition of platforms selected from a continuum of design points along axes of capability, energy requirements, size, and price. Small, numerous, cheap, disposable nodes can be used more effectively by also populating the network with some larger, faster, and more expensive hardware. Smaller devices will trade functionality and flexibility for smaller form factor, lower price, and better energy efficiency.

An analogy can be made to the memory hierarchy commonly found in desktop computer systems. Modern microprocessors typically have expensive and fast on-chip cache, backed by slower but larger L2 cache, main memory, and ultimately on-disk swap space. This organization, combined with a tendency in computation for locality of reference, results in a memory system that appears to be as large and as cheap per byte as the swap space, but as fast as the on-chip cache memory. In sensor networks, where localized algorithms are a primary design goal, similar benefits are possible by using using a heterogeneous spectrum of hardware. A tiered network may seem to be as cheap, portable, disposable, embeddable and energy-efficient as the tiniest nodes, while appearing to have larger nodes' larger storage capacity, higher-speed computation, and higher-bandwidth communication.

To date, much of the research and system construction has been based on a two- or three-tiered architecture. The highest tier is typically a connection to the Internet, where sensor networks can merge with traditional wired, server-based computing. At the smallest end, currently, are platforms such as TinyOS HSW$^+$00 on the Berkeley Mote KKP99. Motes have minimal storage and computation capacity, but have integrated sensors, are small enough to embed in the environment unobtrusively, and use energy slowly enough to run from small batteries. In between these two tiers are "micro-servers"—computers with power on the order of a PDA, running from large batteries or a solar panel, and running a traditional operating system such as Linux.

Routing and in-network processing

Routing is a topic that arises almost immediately in any network as soon as it is large enough to require multiple hops—that is, if there is a pair of nodes that are not directly interconnected. In sensor networks, as in the Internet, this is of course the case. However, there is an important difference in the routing used by sensor networks. The Internet, and much of the earlier research in ad-hoc wireless networks, was focused on building the network as a *transport mechanism*—that is, a way to route packets to a particular endpoint. In a sensor network, efficiency demands that we do as much in-network processing (e.g., data reduction) as possible. Instead of blindly routing packets to a far-away endpoint, many applications do processing *at each hop* inside the network—aggregating similar data, filtering redundant information, and so forth.

For example, emerging designs allow users to task the network with a high-level query such as "notify me when a large region experiences a temperature over 100 degrees" or "report the location where the following bird call is heard" HSI$^+$01; MFHH02. If a node can correlate an incoming audio stream to the desired pattern *locally*, and report only the time and location of a match, the system will be many orders of magnitude more efficient than one that transmits the complete time-series of sampled audio.

The routing necessary to facilitate these queries is not simply end-to-end routing by node address. Routing must often be integrated with and influenced by the application, in stark contrast to Internet-style routing where the two are nearly always separated.

One early example of such a routing system is *Directed Diffusion* HSI$^+$01. Unlike Internet routing, which uses node end-points, Directed Diffusion is data-centric. Data generated by sensor nodes are identified by attribute-value pairs. "Sinks," or nodes that request data, send "interests" into the network. Data generated by "source" nodes that match these interests "flow" toward the sinks. At each intermediate node, user software is given the opportunity to inspect and manipulate the data before it is transmitted to the next hop. This allows application-specific processing inside the network, but also places an additional burden on application developers. Unlike the Internet, where routing is an abstraction that can be essentially ignored, the hop-by-hop routing in Directed Diffusion requires software that is aware of, and perhaps even influences, the routing topology.

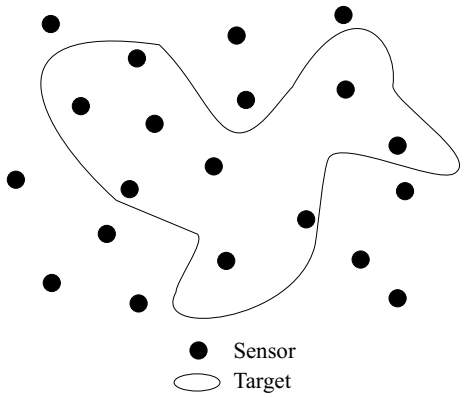

Figure 1.1. Individually, sensors may only be capable of a binary decision: they are within the sensed phenomenon, or they are not. If the sensor positions are known, integration of information from the entire field allows the network to deduce the *size* and *shape* of the target, even though it has no "size" or "shape" sensors.

Automatic localization and time synchronization

Some of the most powerful benefits of a distributed network are due to the integration of information gleaned from multiple sensors into a larger world-view not detectable by any single sensor alone. For example, consider a sensor network whose goal is to detect a stationary phenomenon P, such as that depicted in Figure 1.1. P might be a region of the water table that has been polluted, within a field of chemical sensors. Each individual sensor might be very simple, capable only of measuring chemical concentration and thereby detecting whether or not it is within P. However, by fusing the data from all sensors, *combined with knowledge about the sensors' positions*, the complete network can describe more than just a set of locations covered by P: it can also compute P's size, shape, speed, and so forth. The whole of information has become greater than the sum of the parts: the network can deduce the size and shape of P even though it does not have a "size" or "shape" sensor.

Nearly every sensor network does this type of data fusion, but it is only possible if the sensors have known positions. Positions may be absolute (latitude and longitude), relative (20 meters away from the other node), or logical (inside the barn vs. on the pasture). But, in any of these cases, sensor networks need *automatic localization*. That is, nodes require the capability of localizing themselves after they have been deployed. This is necessary both due to the large scale of deployments, making man-

ual surveys impractical, and due to dynamics that can change a node's position after its initial deployment.

For any phenomenon that is time-varying or mobile, time synchronization is also a crucial service necessary to combine the observations of multiple sensors with each other. For example, synchronized time is needed to integrate multiple position estimates into a velocity estimate.

In some cases, the Global Positioning System (GPS) can and does provide nodes with both their position and a global clock. However, it requires line of sight to several satellites, which is not available inside of buildings, beneath dense foliage, underwater, when jammed by an enemy, or during Mars exploration. In addition, in many contexts, GPS receivers are too expensive in terms of their energy consumption, size, or cost. GPS may not be practical, for example, in a network made up entirely of nodes on the scale of a dust-sized mote, or even the Berkeley COTS mote.

A number of researchers have developed schemes for automatic, ad-hoc localization. Localization schemes often use acoustic time-of-flight ranging GE01; SHS01; WC02 or RF connectivity to beacons of known position BHET03; PSZ01. Many time-of-flight measurement schemes incorporate various forms of sensor network time synchronization EGE02; GKS03.

Distributed Signal Processing

For decades, the signal processing community has devoted much research attention to seamless integration of signals from multiple sources, and sources with heterogeneous sensing modalities. The signal processing literature sometimes refers to this as *array processing*; with heterogeneous sensors, it is often called *data fusion*. There are many applications, such as signal enhancement (noise reduction), source localization, process control, and source coding. It would seem to be a natural match to implement such algorithms in distributed sensor networks, and there has been great interest in doing so. However, much of the extensive prior art in the field assumes *centralized sensor fusion*. That is, even if the sensors gathering data are physically distributed, they are often assumed to be wired into a single processor.

Centralized processing makes a number of assumptions that are violated in sensor networks. For example, in a centralized processor, data can be shared among sensors at (effectively) infinite bandwidth. In sensor networks, communication is expensive in terms of energy, limited in bandwidth, nontrivial in its routing, and unreliable on both short and long timescales. A centralized fusion point also assumes implicit

time synchronization—sensor channels sampled by the same processor also share a common timebase. In sensor networks, distributed sensors are on independent nodes with independent clocks; time synchronization must be made explicit.

In sensor networks, signal processing is a crucial building block. It is, for example, responsible for target tracking, signal identification and classification, localization, and beam-forming. All of these are processes that resolve low-level sensor signals (e.g., acoustic or seismic) into higher level sensors (e.g., "car" or "earthquake"). In sensor networks, the challenge is get the best signal processing results given the bandwidth and computational constraints of the sensing platform.

Storage, search and retrieval

Sensor networks can produce a large volume of raw data—a continuous time-series of observations over all points in space covered by the network. In wired sensor networks, that mass of data is typically aggregated in a large, centralized database, where it later processed, queried and searched. The easiest path toward the adoption of new wireless sensor networks might be to provide users with a familiar model. For example, we might conceive a sensor network with a well-known declarative query interface such as SQL, allowing them to exploit traditional data mining techniques to extract interesting features or event information from the data. However, standard database assumptions about resource constraints, characteristics of data sources, reliability and availability no longer hold in a sensor network context, requiring significant modifications to existing techniques.

Resource constraints introduce possibly the most fundamental difference compared to a traditional database approach. Data mining over a massively distributed database which is under energy, bandwidth and storage constraints is daunting. The data cannot be transmitted to and stored at a central repository without adversely impacting lifetime of the network since limited energy resources are available at each node. An alternative approach could be to store data locally at sensor nodes, and query such data on-demand from users. However, the need to operate unattended for many years, coupled with cost and form factor constraints on sensor nodes, limits the amount of storage available on nodes.

In addition to storage constraints, processing and memory constraints on sensor nodes, especially on the low-end such as the Berkeley Motes, adds a dimension to optimize in such systems. The need for in-network processing necessitates that each sensor node act as a data processing

engine, an implementation of which is challenging on nodes that function under tight CPU and memory constraints.

Traditional databases are not suitable when the data source is continuous and needs to be processed in real time, which might be required in some sensor network applications. Typical query processing deals with stored data on disk, rather than streaming sensor data. New techniques will be required for online processing of data streams that do not assume availability of significant secondary storage or processing power.

Reliability and availability of sensor nodes need to be considered in the construction of data processing engines. Since sensor data is transmitted over wireless connections with varying channel characteristics, the database should be able to handle variable delay and different delivery rates from different nodes. Further, these delays should be handled with low energy overhead—that is, minimal listening duration over the radio.

Actuation

In many cases, a sensor network is an entirely passive system, capable of detecting the state of the environment, but unable to change it or the network's relationship to it. *Actuation* can dramatically extend the capabilities of a network in two ways. First, actuation can enhance the sensing task, by pointing cameras, aiming antennae, or repositioning sensors. Second, actuation can *affect* the environment—opening valves, emitting sounds, or strengthening beams BCY+98.

One of the most commonly described forms of actuation is sensor mobility. For example, Sibley *et al.* developed a prototype "RoboMote," which brings autonomous mobility to the Berkeley Mote sensor platform SRS02. They propose a number of ways to leverage such mobile sensors, including *actuated boundary detection*. The size and shape of a target region can be determined with more precision if the sensors can move, as shown in Figure 1.2. Others have proposed using mobile nodes to harvest energy from the environment, delivering it to stationary sensors RSS+03; PB99.

Mobility is also central to the DARPA SHM program (AT. In that system, nodes collaboratively construct a map of their relative positions using acoustic localization. Each node is equipped with a mobility subsystem consisting of small rocket motors. The goal is to keep the target area covered by sensors, even as nodes fail or are moved. If the sensors detect that a gap in the field has opened, nearby nodes fire their rockets and move in to fill the open space.

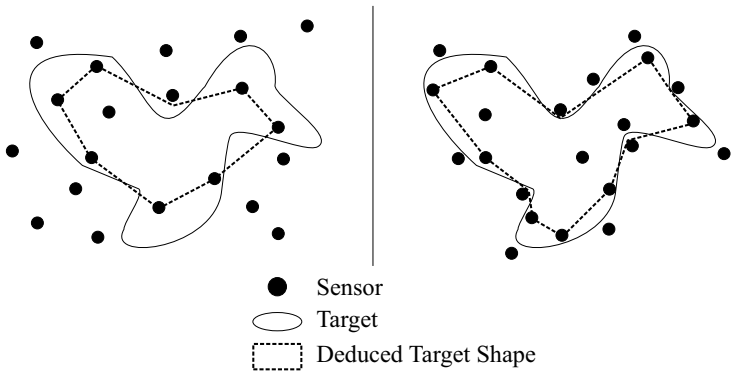

Figure 1.2. left) A sensor field might deduce the shape of a target area (e.g., contaminated soil) by drawing a convex hull around the sensors that are within the contamination, with concave deviations around uncontaminated sensors. *right*) If the sensors are *mobile*, they can collaboratively adjust their positions, estimating the shape of the target with greater accuracy.

Simulation, monitoring, and debugging

Simulation and debugging environments are important in any large-scale software development project. In sensor networks, a number of factors make the use of innovative development models particularly important.

Large-scale sensor networks will not be able to simply send all the raw sensor data back to a centralized recording point. Energy and channel capacity constraints will require as much localized processing as possible, delivering only a (small) high-level sensing result. As we described earlier, a perfect system will reduce as much data as possible as early as possible, rather than incur the energy expense of transmitting raw sensor values further along the path to the user.

For a system designer, there is an unfortunate paradox intrinsic to this ideal: the data that must be discarded to meet the energy and channel capacity constraints are necessary for the evaluation and debugging of the data reduction process itself. How can a designer evaluate a system where, by definition, the information necessary for the evaluation is not available? That is, how can we be sure that the final, high-level sensing result delivered by the system is an accurate reflection of the state of the environment—when sensor networks are, by definition, deployed where we have insufficient energy and channel capacity to record all the raw data?

This fundamental problem makes simulation crucial in sensor networks. A simulator may be the only environment in which a sensor net-

work can both run at a large scale and record each raw datum. This allows the "environment" as observed by the sensors to be reconciled with the high-level, distilled result delivered by the software being tested. Several example of these systems exist, including TOSSIM LLWC03, EmStar EBB+03, and sensor network extensions to GloMoSim and ns-2.

Security and Privacy

In sensor networks, many unique challenges arise in ensuring the security of sensor nodes and the data they generate PSW+01; EG02; LEH03. For example, the fact that sensors are embedded in the environment presents a problem: the physical security of the nodes making up the network can not be assured. This can make security significantly different than in Internet servers. In sensor networks, attackers may modify node hardware, replace it with malicious counterparts, or fool sensors into making observations that do not accurately reflect the environment. To a single temperature sensor, a small match may look no different than a forest fire. Algorithms for ensuring network-wide agreement are crucial to detecting attacks because we can no longer assume the security of individual nodes, or the data they generate.

The limited resources on the smallest sensor nodes also can pose challenges. Many encryption schemes are impractically resource-intensive, consuming far more energy, memory, and computational time than would be required to send a raw, unprotected data value. In addition, protection of data from eavsedropping en-route—particularly important in wireless networks—traditionally implies end-to-end encryption. That is, data is encrypted as soon as it is created, transmitted through the network, and finally received by a secured server where decryption keys can be stored without danger of exposure. Unfortunately, finite energy drives the need for in-network processing in sensor networks, which confounds the traditional security schemes. Nodes inside the network can not perform application-specific processing on encrypted data. Decrypting the data at each node implies decryption keys are stored at each node; unfortunately, the node hardware itself is exposed, and can not be assumed to be out of reach of attackers.

As with many types of information technology throughout history, sensor networks also raise important questions about the privacy of individuals. Certain aspects of privacy have gradually eroded due to various forces—for example, the tracks we leave behind by using credit, the ubiquity of surveillance cameras, and the seeming omniscience of Internet search engines. Sensor networking, similarly, is a technology that

can be used to enrich and improve our lives, or turned into an invasive tool. As sensor networks become more widespread, they will become an important point to consider in the continuing debate between public information and private lives.

6. Conclusions

Recent advances in miniaturization and low-cost, low-power electronics have led to active research in large-scale networks of small, wireless, low-power sensors and actuators. Pervasive sensing that is freed from the burden of infrastructure will revolutionize the way we observe the world around us. Sensors networks will automatically warn us of invisible hazards—contaminants in the air we breathe or the water we drink, or far-away earthquakes soon to arrive. Sensor networks can open the eyes of a new generation of scientists to phenomena never before observable, paving the way to new eras of understanding in the natural sciences.

Sensor networks will eventually be integral to our homes and everyday lives in ways that are difficult to imagine today. Computers themselves were once specialized tools limited to the esoteric domain of rote mathematical computation. It was inconceivable at the time that people would want computers in their homes, yet in the span of a single generation they have become household fixtures. Perhaps, someday, electronic sensing will be as natural to us as our own innate senses. Only time will tell.

Acknowledgements

This chapter was supported by the National Science Foundation, via the Center for Embedded Networked Sensing (CENS), NSF Cooperative Agreement #CCR-0120778. Additional support was provided by Intel Corporation. The authors are grateful for contributions by Deepak Ganesan, Lewis Girod, and Michael Klein.

References

DARPA Advanced Technology Office (ATO). Self-healing minefield. http://www.darpa.mil/ato/prog

A.A. Berlin, J.G. Chase, M.H. Yim, B.J. Maclean, M. Oliver, and S.C. Jacobsen. MEMS-based control of structural dynamic instability. *Journal of Intelligent Material Systems and Structures*, 9(7):574–586, 1998.

Nirupama Bulusu, John Heidemann, Deborah Estrin, and Tommy Tran. Self-configuring localization systems: Design and experimental evaluation. *ACM Transactions on Embedded Computing Systems*, page To appear., May 2003.

Jeremy Elson, Solomon Bien, Naim Busek, Vladimir Bychkovskiy, Alberto Cerpa, Deepak Ganesan, Lewis Girod, Ben Greenstein, Tom Schoellhammer, Thanos Stathopoulos, and Deborah Estrin. EmStar: An environment for developing wireless embedded systems software. Technical Report 0009, Center for Embedded Networked Sensing, University of California, Los Angeles, March 2003. http://lecs.cs.ucla.edu/publications.

Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In Vijay Atlury, editor, *Proceedings of the 9th ACM Conference on Computer and Communication Security (CCS-02)*, pages 41–47, New York, November 18–22 2002. ACM Press.

Jeremy Elson, Lewis Girod, and Deborah Estrin. Fine-grained network time synchronization using reference broadcasts. In *Proceedings of the 5th ACM Symposium on Operating System Design and Implementation (OSDI-02)*, Operating Systems Review, pages 147–164, New York, December 9–11 2002. ACM Press.

National Center for Atmospheric Research. Cyberinfrastructure for environmental research and education. Technical report, Sponsored by the National Science Foundation, September 2003. http://www.ncar.ucar.edu/cyber/.

Lewis Girod and Deborah Estrin. Robust range estimation using acoustic and multimodal sensing. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2001)*, March 2001.

S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, Los Angeles, November 2003.

John Heidemann, Fabio Silva, Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, and Deepak Ganesan. Building efficient wireless sensor networks with low-level naming. In *Proceedings of the Symposium on Operating Systems Principles*, pages 146–159, Chateau Lake Louise, Banff, Alberta, Canada, October 2001. ACM.

Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. In *Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX)*, pages 93–104, Cambridge, MA, USA, November 2000. ACM.

J.M. Kahn, R.H. Katz, and K.S.J. Pister. Next century challenges: mobile networking for Smart Dust. In *Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking*, pages 271–278, 1999.

Y.W. Law, S. Etalle, and P.H. Hartel. Assessing security-critical energy-efficient sensor networks. In *Proceedings of IFIP WG 11.2 Small Systems Security Conference*, Athens, Greece, May 2003.

Philip Levis, Nelson Lee, Matt Welsh, and David Culler. Tossim: Accurate and scalable simulation of entire tinyos applications. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, Los Angeles, November 2003.

Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI)*, pages 131–146, Boston, MA, USA, December 2002.

Gregory J. Pottie and Rodney Brooks. Towards a robotic ecology. DARPA ISAT Study, 1999.

Gregory J. Pottie and William J. Kaiser. Wireless integrated network sensors. *Communications of the ACM*, 43(5):51–58, May 2000.

Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, and J. D. Tygar. SPINS: Security suite for sensor networks. In *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MOBICOM-01)*, pages 189–199, New York, July 16–21 2001. ACM Press.

Maurizio Piaggio, Antonio Sgorbissa, and Renato Zaccaria. Autonomous navigation and localization in service mobile robotics. In *Proceedings*

of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2001), March 2001.

Mohammad Rahimi, Hardik Shah, Gaurav Sukhatme, John Heidemann, and Deborah Estrin. Energy harvesting in mobile sensor networks. Technical report, Center for Embedded Networked Sensing, 2003. http://www.cens.ucla.edu/Proj Descriptions/energyharvesting/EnergyHarvesting.pdf.

Andreas Savvides, Chih-Chien Han, and Mani Srivastava. Dynamic Fine-Grained localization in Ad-Hoc networks of sensors. In *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MOBICOM-01)*, pages 166–179, New York, July 16–21 2001. ACM Press.

Gabriel T. Sibley, Mohammad H. Rahimi, and Gaurav S. Sukhatme. Robomote: A tiny mobile robot platform for large-scale sensor networks. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA2002)*, 2002.

Kamin Whitehouse and David Culler. Calibration as parameter estimation in sensor networks. In *Proceedings of the first ACM International Workshop on Wireless Sensor Networks and Applications*, pages 59–67, 2002.