The Sensor Spectrum: Technology, Trends, and Requirements

Joseph M. Hellerstein, Wei Hong, Samuel R. Madden
UC Berkeley, Intel Research Berkeley, MIT
jmh@cs.berkeley.edu, whong@intel-research.net, madden@csail.mit.edu

Abstract

Though physical sensing instruments have long been used in astronomy, biology, and civil engineering, the recent emergence of wireless sensor networks and RFID has spurred a renaissance in sensor interest in both academia and industry. In this paper, we examine the spectrum of sensing platforms, from billion dollar satellites to tiny RF tags, and discuss the technological differences between them. We show that battery powered sensor networks, with low-power multihop radios and low-cost processors, occupy a sweet spot in this spectrum that is rife with opportunity for novel database research. We briefly summarize some of our research work in this space and present a number of examples of interesting sensor network-related problems that the database community is uniquely equipped to address.

1 Introduction

Sensor networks have attracted a tremendous amount of attention and media publicity from both the research community and industry. Many universities have started offering courses on sensor networks. A number of academic conferences and workshops have been created to enable publication in the area. Real world sensor networks have been deployed by several research groups [39, 9, 20]. In the industry, many large corporations (e.g., Intel, HP, Accenture, etc.), have started research projects around sensor networks, and a number of startup companies, such as Crossbow, Dust, Ember, and Millennial Net, have been formed in the last few years to sell sensornet hardware and software.

Like many other communities, the database community has been drawn to this emerging space. SIGMOD 2004's Call For Paper explicitly lists "embedded and self-organizing databases" as one of the areas of interest, presumably in direct reference to sensor networks. Research on streaming data, a topic related to sensor networks, is already being widely pursued in the database community today, with major theory and system-building efforts coming out of a number of groups. Most of these endeavors are predicated on the future proliferation of sensor devices, which will output data streams that need to be processed via continuous query engines. However, relatively few of these projects are actively involved with the researchers and engineers developing sensor devices and sensor network technology.

In this paper, we examine a spectrum of sensing technologies, based on ongoing discussions and collaborations with sensor researchers at UC Berkeley and Intel Research, and with affiliated industrial groups. Using the evolution of sensor technology as a guide, we explain why the database community should get excited by sensor networks and illustrate opportunities where it can contribute to this emerging space. We include a summary of some of the initial successes at applying database techniques in this area and present some of the database research challenges that we foresee on the horizaon.

The rest of this paper is organized as follows. In section 2 we present our spectrum sensing of scenarios and technology, discussing the differences in size, scale, and capabilities of different platforms for remote data collection and sensing. Section 3 then explains why recent changes in the sensor spectrum make database research more applicable than ever to this space and presents several research problems we believe to be particularly worthwhile given the novel challenges of sensor networks and skill set of the database community. Finally, Section 4 summarizes the lessons of the paper and concludes.

2 A Spectrum of Sensing Technology

We begin by examining a range of sensing scenarios, starting with several more traditional sensing and sensor fusion applications and moving towards the new generation of sensing applications that has arisen with the advent of cheap, low cost sensing hardware and radios.

Figure 1 illustrates this spectrum, with the four major areas we discuss shown along it. We begin at the left of the spectrum, with satellites, and then discuss data-loggers, RFID tags, and finally, wireless sensor networks, which, as we shall see, lie at a sweet-spot in terms of computational power, cost, and size that make them especially attractive to the database research community.

2.1 Remote Sensing via Satellite

Some of the highest-volume sensing comes from earth-orbiting satellites like NASA's EOSDIS (Earth Observing System Data and Information System) project. The database research community attempted to aggressively address satellite-based remote sensing in the early 1990's [21, 16, 3]. Unfortunately, DBMS-centric approaches did not have much impact in this arena, for several reasons. First, the volume of structured data in some satellite applications is dwarfed by the volume of so-called "large objects" – large raster images. In these cases, the DBMS only really helps with the low-volume metadata, which some consider to be secondary to the "real", high-volume image data.

This explanation does not tell the whole story, however. In fact, some of the most important EOSDIS data (e.g., the CERES data on radiation [41]) is structured and very "database-like". Moreover, a team led by Mike Stonebraker

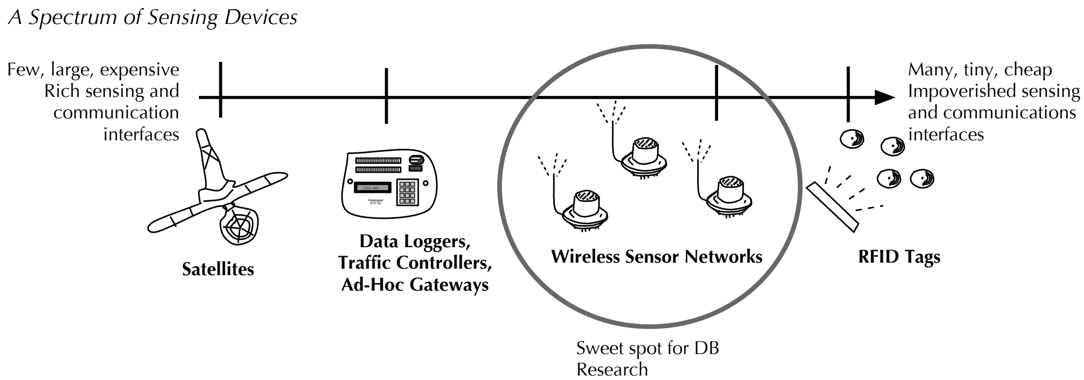

Figure 1: A spectrum of sensing devices, from large, expensive satellites, to tiny, cheap RFID tags. In this paper, we argue that wireless sensor networks are at a sweet-spot in terms of computational power, communication interfaces, and cost, such that a number of interesting research issues arise that are not present at other points in the spectrum.

and Jim Gray proposed an alternative architecture for EOS-DIS [19] that argued persuasively in economic and technical terms for a DBMS-centric approach based on off-the-shelf object-relational software. So why were database techniques not adopted for satellite data?

The answer lies with the "market" for remote sensing. NASA is one of the only customers for remote sensing data management software. They best understand their own needs, and their software budgets are fairly large – software is cheaper than launching a set of satellites. Hence the traditional DBMS focus on general-purpose applicability and flexibility is not of primary importance to this community. Instead, they seem to prefer to write or contract out custom code. As of 2001, their Product Generation Executive codebase, which does their data processing, was over 750,000 lines of code, with an additional 550,000 lines of validation code [34]. This compares to the size of a significant DBMS implementation – the open-source PostgreSQL codebase is currently 412,000 lines of code, for example. If the sole "customer" is willing to write that much custom code, this is probably not an attractive space for clever general-purpose systems research.

2.2 Dataloggers

Field scientists, particularly biologists and environmental scientists, have long used dataloggers and radio beacons to collect periodic readings about local environmental conditions such as light, temperature or humidity. There are a number of commercial data loggers that are available and widely used in the scientific community [6, 15]. Campbell Scientific [6] is a well known vendor; their data loggers are relatively large instruments (ranging in size from 36 in^3 and weighing 1 pound for small, limited storage devices to 2300 in^3 and 40 pounds for larger devices with batteries for long-term remote operation and significant storage [7]) that can interface to a wide range of sensors and are designed to operate in a number of harsh environments. Their newest generation of data loggers includes interfaces to single-hop, point-to-point radios, and the smallest data loggers feature power consumption characteristics that allow them to run on the equivalent of a pair of AA batteries at low duty cycle (e.g., tens or hundreds of samples a day) for months at a time. Larger, more power hungry versions that require large solar cells or a wall-socket offer significantly faster sampling rates and high-precision analog-to-digital converters.

Most data-loggers collect readings periodically or according to a scheduled "program" – a set of pre-specified time periods throughout the day. In addition to basic logging facilities, these data loggers support time averaging and the tracking of windowed minima/maxima. In most settings, users are expected to buy a few at a high premium (thousands of dollars apiece, once the cost of sensors and radios is figured in – see, for example, the Onset Computer Pricing Page [14]) to monitor a few locations.

Due to the relatively small number of loggers, low data rates, simple schemas, and simple (or non-existent) communication technologies involved in these applications, they have not received much attention from the systems community, as we discuss in more detail in Section 2.5.

2.3 Traffic Monitoring

Another traditional application of sensor technology is traffic sensing. In California, the California Department of Transportation (CalTrans) has instrumented freeways with approximately ten-thousand inductive loop sensors. These sensors consist of metal loops that are cut into the freeway and have a small current induced in them when a vehicle passes overhead. These loops are connected to a loop controller, such as the CalTrans Model 170 Controller [25] in a large metal cabinet on the side of the freeway, which aggregates data into vehicle counts and speeds and relays data back to a central server via a modem or wireless link. The Washington State Department of Transportation website [49] illustrates the design of a sensing system at a single freeway interchange.

There continues to be a large amount of research associated with these sensors in the traffic, civil, and electrical engineering communities; see, for example, [35, 42, 48]. This research, for the most part, appears to focus on two major areas: improving the resolving ability and quality of the sensors themselves, or doing more sophisticated offline analysis of collected sensor data. The complexity of the collected data is low: each loop produces records with identical schemas, and on-freeway aggregation limits the rate at which it is produced. Furthermore, the infrastructure to load these records into traditional database and data analysis packages has been long established, making innovation difficult of little perceived utility.

2.4 RFID

The prior sections described fairly mature technologies. RFID tags represent the low end of emerging sensor deployments, in terms of both technology and cost. RFID tags are small enough to embed in a standard price tag, and cost only pennies per unit. An RFID tag is capable of transmitting a value (e.g. an ID) and perhaps computing a handful of instructions (e.g. decrement a counter) when brought in proximity to an RFID reader. By contrast, RFID *readers* are much more expensive (currently from hundreds to thousands of dollars, though prices are falling) and they require significant power to run. Hence RFID readers are typically immobile, connected to a fixed power source. RFID is already in use in a number of applications, including highway toll systems and asset tracking for supply chain and retail applications.

Because RFID is low-tech and already quite inexpensive, it is likely to be a source of "buzz" in the short run. However, the requirement of physical proximity to a reader makes RFID a fairly limited technology; it is essentially one step more sophisticated than bar-codes. The refinements offered by RFID are likely to increase database insertion rates beyond what is seen in high-end retail data warehousing.

2.5 Ad-hoc mobile networks

At the other end of the emerging sensor data space are networks of large, desktop-PC class devices that are battery-powered and radio-equipped, and are currently about 120 in^3. Common uses for these platforms are in areas such as telematics for automobiles and military and anti-terrorism applications. Such applications are CPU and data intensive and require hardware that consumes roughly as much energy as a laptop PC, but can incorporate sophisticated sensing and signal analysis.

A popular ad-hoc, mobile node is Sensoria Corporation's Linux-based "sGate" [45]. The sGate weighs about 3 pounds without batteries, incorporates a 300 Mhz processor, 64 MB of RAM and has draws of 10 watts power, on average (versus about 40 watts for a typical laptop.) It can be battery powered for "up to 24 hours" using batteries that weigh several pounds. It includes a 10 or 100 mW 2.4 Ghz radio.

Though there are a number of interesting ad-hoc network issues presented by these kinds of mobile nodes, mobile databases have been a popular research topic since the early nineties [30, 1]; the research issues associated with querying these ad-hoc network gateways are similar.

2.6 Wireless Sensor Networks

Between RFID and these large gateways are wireless sensor networks. The devices in these networks typically consist of small, power-efficient, battery-powered nodes with low-range radios and low-cost sensors. A typical class of devices are the Berkeley *Motes*, which incorporate an 8-bit, 7Mhz processor, a ChipCon CC1000 radio with a range of about 100 feet, 4KB of RAM, 128KB of program memory, and 512KB of off-chip non-volatile EEPROM. Motes accept a variety of *sensor boards* that provide the ability to sense magnetic field, vibration, temperature, light, heat, humidity, air-pressure, and other environmental attributes. Figure 2 shows a diagram of a Berkeley "Mica" mote, next to a sensor board with light, temperature, magnetic field, and acceleration sensors, as well as a piezo-electric buzzer. Such motes can be purchased from Crossbow Technology [18] for about $200.00, including a sensor board; prices have been dropping steadily since these devices were introduced about two years ago. Projections from one vendor [24] suggest that in the next two years, this class of devices should shrink in size to about 2 cubic mm (without batteries), and in cost to about $10.

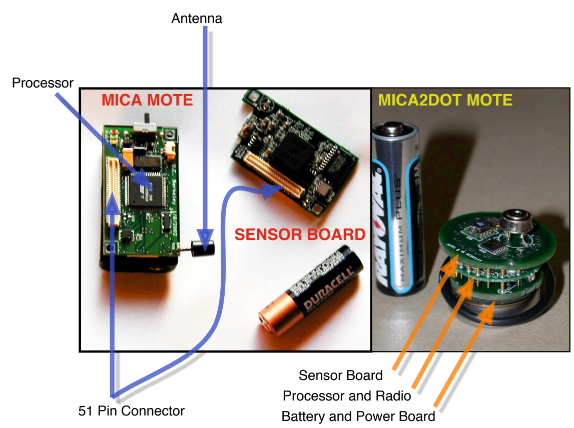

Figure 2: Berkeley Mica Mote

Though these motes appear at first to be only small, cheap versions of the ad-hoc nodes described above, they enable a surprisingly large class of new applications. Their low-cost and ease-of-deployment mean that they can be placed in spatially dense configurations throughout an area. The networking software that runs on these devices allows them to self-assemble into ad-hoc networks, such that data can be relayed across multiple hops and from long distances. The operating system software, called TinyOS [28], has been designed from the ground up with power-efficient sensing in mind, meaning that intelligent applications can easily run for months at a time off batteries that provide half the capacity of a AA cell.

Although the technology is in its infancy, several scientific users are already seeing major gains in the field from these devices. For example, we recently deployed a network of these devices for purposes of monitoring microclimates around redwood trees in the UC Berkeley Botanical Garden. The biologists involved are interested in the ways in which the trees take an active role in affecting temperature and humidity in the canopy around them[20]. Using a network of 30 motes on two trees, collecting readings every few minutes, we have been able to precisely illustrate how humidity and temperature varies at different levels of the trees. As we extend the network with more sensors on more trees, we hope to observe other effects, such as the ways in which these environmental properties vary in different parts of the forest.

Other recent sensor network applications include tracking vehicles within a field of these sensors [44, 12], monitoring bird habitats on Great Duck Island, off the coast of Maine [39], monitoring vineyards [5], tracking animals in a wildlife refuge [33], and condition-based maintenance of fab equipments at Intel. None of these applications would have been possible with RFID or bulky, expensive nodes like Sensoria's mGate.

2.7 Predictions from Moore's Law

Many of the challenges of designing software for sensor networks have to do with the limited resources available in today's sensor network hardware. In particular, processor speed, RAM, radio bandwidth, and energy capacity are all severely constrained on hardware like the Berkeley motes.

Moore's law suggests that memory density and processor speed will continue to grow at an exponential rate: in ten years, devices as large as a mote will have the processing power and storage of today's server-class machines. However, neither energy density nor the energy costs of communication are expected to scale in this fashion; over the past ten years the energy density of commercially available batteries has changed only modestly – expensive, "high capacity" Lithium batteries sold for cameras and phones have only marginally greater energy densities than their alkaline counterparts [22, 23]; the real advantage of these batteries lies in extended lifetime under high-current drains such as charging the flash on a camera.

Similarly, radio bandwidth is not expected to scale as dramatically as processor speed or RAM capacity. According to the Berkeley Wireless Radio Center's (BWRC) PicoRadio project, it should eventually (at some unspecified time in the future) be possible to build a 100μW radio transceiver (at 10% duty cycle), with a range of about 10 meters [43]. The current mote radio uses about 15mW when running in the 10m range, [13] when operating at 100% duty cycle, or about 1.5 mW at a 10% duty cycle. Thus, the difference in power consumption between a cheap, commercially available radio and the best the research community can conceive is about an order of magnitude. Though this reduction is significant, it is not enough to eliminate concerns associated with the energy costs of communication, particularly given that the PicoRadio predictions hold only in low-channel-utilization, low-range environments and have yet to be demonstrated in practice.

Thus, sensor networks will continue to be bandwidth and energy limited in the foreseeable future, so the research community needs to focus particularly on the challenges surrounding these resources.

3 The Emerging DB Research

Automated sensing is not new technology, but to date sensors have not been a big driver of database research. Looking at our spectrum of devices above, we see two trends that are bringing sensor-data support to the fore.

The first is the decreased cost and size of the actual sensing hardware. Prior generations of data-intensive sensors were very expensive (as in the case of EOSDIS) and/or burdensome to use (as in the case of large dataloggers). As a result, these sensors were not widely deployed, and the interested customers at the time preferred to buy or write application-specific software. As sensors become much smaller and cheaper, they will be more widely used, and general-purpose off-the-shelf analysis software will become much important in this domain.

The second, more siginificant trend is the convergence of sensing devices and networked computing in a single package. Once sensors are bundled with a processor and a radio, it becomes very easy to deploy large quantities of sensors that set up their own ad-hoc network infrastructure.

These trends have led the database community to begin a number of research agendas related to the rise of sensor technology. We review them here, and place them in context of our sensor technology spectrum.

3.1 A Database Language for Sensor Tasking?

The long-range vision for sensor networks is that they will be deployed in very large quantities – somewhere between many thousands and billions, depending on the prognostication. This scale requires that the programming and and deployment of these devices be extremely simple, and amenable to significant adaptivity in the field. Networking researchers have focused on routing and network scheduling issues in this regard. An open question, however, is how collections of these sensors can be programmed or "tasked" to perform a family of different behaviors. There are a number of approaches suggested for this problem, some of which are based on database languages.

First, we note that Turing-complete sensor programming languages are unlikely to be widely used. Sensor programming entails simultaneously dealing with the challenges of distributed and embedded programming. Both are difficult; the multiplication of the difficulties is prohibitive for all but a small number of experts.

As one set of alternatives, various parties have proposed declarative query languages for sensor networks. There are two main approaches currently being advocated. The first installs an independent query at each node, with the multiple query instances working together to achieve some goal; Directed Diffusion [32] is the best-known example of this idea. The second approach uses SQL-style, *collective* querying of the entire sensor network as a single streaming database [50, 36]. Both of these approaches shield the user from execution details, but they differ in their degree of abstraction. The per-node diffusion approach requires users to reason about the interactions of multiple query instances, while the collective query approach lets users state their overall semantics on a high-level task for the entire sensor field. The collective approach also allows for various cross-node optimizations that have been the subject of recent research [36, 37].

Our own TinyDB work is a collective query model. It is being shipped by Crossbow, the commercial vendor of the Berkeley motes, largely because they believe the programming interface to be more broadly appealing than programming TinyOS directly.

While the streaming database metaphor is a rosy story for many scenarios, we note that some natural sensor tasks are difficult to express declaratively. For example, vehicle tracking is an inherently localized task. The current TinyDB approach to tracking is based on active database rules [27], which look more like diffusion than collective queries. Rule programming is notoriously tricky when scaling beyond a handful of interacting rules.

An alternative to a declarative, calculus-based language is an algebraic, dataflow language. This could be used both in per-node or collective metaphors, and is equally amenable to automatic logical optimization. More flexible yet would be a generic "boxes-and-arrows" dataflow language over *arbitrary* code modules developed in a low-level language. This is the approach taken by various data analysis and visualization tools in the scientific and financial data mining domains (e.g. [31, 47], etc.) If the semantics of the modules are arbitrary, opportunities for automatic optimization become limited.

Of course, some extensible middle ground is also possible for both declarative and algebraic languages; TinyDB's language supports user-defined aggregates and scalar functions, for example, with extensibility APIs enabling various optimizations [36].

3.2 Declarative Acquisition

Aside from choosing the appropriate language and programming abstractions, data-acquisition – the capturing of samples from the sensing hardware on the devices – is one of the fundamental challenges for this class of device. *Acquisitional query processing*(ACQP)[37] has to do with the active role that a declarative query processor for sensor networks must take in choosing when and where data should be captured. Unlike in a traditional database system, which simply processes whatever data happens to be on disk at the time, sensor networks can be configured to produce data at different rates or in response to different events. This reconfigurability is something that can be exploited during query processing to avoid unnecessary utilization of energy in the network and improve the resolution of results returned to the user.

Existing sensor network database research has proposed several simple extensions to declarative languages for controlling data acquisition: most commonly, a *sample rate* is used to specify the periodicity of data acquisition. There are, however, a number of other possible policies for initiating data acquisition, for example:

- **Asynchronous Acquisition:** Rather than collecting data at a regular rate, samples could be acquired when some condition is met or some external event occurs, as described in our paper on ACQP[37]. Open problems in this space include understanding the appropriate language abstractions for specifying asynchronous rules and building an asynchronous query execution engine that is both low-power and able to quickly respond to external events. These two challenges are closely related since the expressiveness of the language must be sufficiently constrained to allow efficient execution.
- **Lifetime-based acquisition:** The ACQP paper also discusses the possibility of allowing users to specify a desired network lifetime rather than a periodic sample rate. Given a lifetime, sensors in the network would adjust their sample rate based on available power and rate of power consumption over time. Unfortunately, the ACQP paper does not go into great detail about how this could be implemented, except in the simplest case where a node is forwarding data over a single network hop (e.g. is doing no additional communication besides its own.)

  The challenge of lifetime optimization is that it has a number of different possible definitions, some of which cause it to be a whole-network property that is very difficult to optimize for. Possible definitions of lifetime include: time until any sensor fails, the time until all sensors fail, or something in-between. In the first case, to maximize sample rates and meet lifetime goals, sensors will need to dynamically adjust the network topology to reduce the burden of message forwarding on nodes with little power and shed operator load so that all sensors' batteries drain at the same rate (or as close to the same rate as possible.)

3.3 Managing Power in Large Networks of Tiny Devices

Given that power is one of the fundamental constraints in these networks, it is interesting to consider how power plays into query execution. First of all, because communication is orders of magnitude more expensive than local computation, it is much more efficient to move the data processing logic into the network rather than collecting and processing sensor data outside the network. There are two issues here: first, the high-level, physical operator placement issues regarding where various operators should run to reduce excessive communication and minimize power consumption, and second, the low-level systems issues about how to make operators run efficiently and how to schedule computation and communication such that devices use as little energy as possible.

There has been some work on operator placement [4], but a number of challenges still remain. Particularly interesting are issues pertaining to heterogeneity – that is, choosing where to place an operator given that the nodes in the network may have differing processing power or battery life, and may be experiencing different computational or communications loads over load. Dealing with dynamic heterogeneity, such as variations in load, suggests some form of adaptive query optimization, such as eddies[2].

4 Conclusions

Although the database community has put some effort towards managing data generated from various older sensing technology, the trend towards large wireless sensor networks consisting of hundreds or thousands of tiny, low-power, and low-bandwidth smart sensor devices presents a number of exciting new database challenges. To make sensor networks a ubiquitous technology, the database community must collaborate with sensor network researchers from other fields to design unique solutions to manage and process data both inside and outside of sensor networks.

At Berkeley, our sensor network database research has benefited tremendously through close collaboration with sensor network researchers and real world users of sensors who are trying to tackle some of the challenges raised in this paper. Though TinyDB [38] has enjoyed some initial successes in both the research community and the sensor network users community, we have only scratched the surface of the diverse set of problems in this space. We look forward to the day when more database researchers become involved in this rapidly emerging and exciting research area.

References

[1] R. Alonso and S. Ganguly. Query optimization in mobile environments. In *Workshop on Foundations of Models and Languages for Data and Objects*, pages 1–17, September 1993.

[2] R. Avnur and J. M. Hellerstein. Eddies: Continuously adaptive query processing. In *Proceedings of the ACM SIGMOD*, pages 261–272, Dallas, TX, May 2000.

[3] T. S. . S. Benchmark. Michael stonebraker and jim frew and kenn gardels and jeff meredith. In *Proceedings of SIGMOD*, pages 2–11, June 1993.

[4] B. Bonfils and P. Bonnet. Adaptive and decentralized operator placement for in-network query processing. In *Proceedings of the First Workshop on Information Processing in Sensor Networks (IPSN)*, April 2003.

[5] T. Brooke and J. Burrell. From ethnography to design in a vineyard. In *Proceeedings of the Design User Experiences (DUX) Conference*, June 2003. Case Study.

[6] Campbell Scientific. Campbell Scientific Inc. Measurement and Control Systems. Web Page. `http://www.capbellsci.com`.

[7] Campbell Scientific, Inc. Datalogger comparison table. Technical report, 2003. `http://www.campbellscientific.com/compare.html`.

[9] A. Cerpa, J. Elson, D.Estrin, L. Girod, M. Hamilton, and J. Zhao. Habitat monitoring: Application driver for wireless communications technology. In *ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, 2001.

[12] M. Chu, H. Haussecker, and F. Zhao. Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. In *International Journal of High Performance Computing Applications, 2002.*, 2002.

[13] C. Corporation. *CC1000 Single Chip Very Low Power RF Transceiver Datasheet*. `http://www.chipcon.com`.

[14] O. C. Corporation. HOBO/StowAway Data Loggers and Shuttles. Web Site, September 2003. `http://www.onsetcomp.com/Commerce_Pages/product.ccml?6,0,ck0365980adb90ef33a9cf632748bc4bad,oc2_170836,,,hobo.html`.

[15] O. C. Corporation. Onset computer: Battery-operated data loggers and controllers. Web Page, 2003. `http://www.onsetcomputer.com`.

[16] P. Cowley, B. Farrell, P. Gilna, and L. Russel. Scientific data management: Real-world issues and requirements. SIGMOD Panel and Abstract, June 1992.

[18] Crossbow, Inc. Wireless Sensor Networks (Mica Motes). `http://www.xbow.com/Products/Wireless_Sensor_Networks.htm`.

[19] F. Davis, W. Farrell, J. Gray, R. Meches, R. Moore, S. Sides, and M. Stonebraker. Eosdis alternative architecture. Web Site, January 1995. `http://research.microsoft.com/~gray/EOS_DIS/`.

[20] T. Dawson. Fog in the california redwood forest: ecosystem inputs and use by plants. *Oecologia*, 117(4):476–485, 1998.

[21] J. Dozier. Access to data in nasa's earth observing system. SIGMOD Keynote Address and Abstract, June 1992.

[22] I. Duracell. *MX1500*, September 2003. Available from `http://www.duracell.com/oem/Pdf/MX1500.pdf`.

[23] I. Duracell. *Ultra CR-V3*, September 2003. Available from `http://www.duracell.com/oem/Pdf/CRV3.pdf`.

[24] Dust Inc. Company Web Site. `http://www.dust-inc.com`.

[25] J. Forester. 170 controller. Technical report, California Department of Transportation, 2003. See `http://www.dot.ca.gov/hq/esc/ttsb/electrical/170c.htm`.

[27] J. Hellerstein, W. Hong, S. Madden, and K. Stanek. Beyond average: Towards sophisticated sensing with queries. In *Proceedings of the First Workshop on Information Processing in Sensor Networks (IPSN)*, April 2003.

[28] J. Hill, R. Szewczyk, A. Woo, S. Hollar, and D. C. K. Pister. System architecture directions for networked sensors. In *ASPLOS*, November 2000.

[30] T. Imielinski and B. Badrinath. Querying in highly mobile distributed environments. In *VLDB*, Vancouver, Canada, 1992.

[31] K. Inc. Khoral Inc: Khoros Pro 2001 Integrated Development Environment. Web Page. `http://www.khoral.com/khoros/`.

[32] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *MobiCOM*, Boston, MA, August 2000.

[33] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Pehand, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proceedings of the Tenth ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, October 2002.

[34] J. Kibler. CERES Science Team Meeting. Technical report, NASA Langley Research Center, May 2001. Available from `http://asd-www.larc.nasa.gov/ceres/science_team/ST_Mtg.0105.pdf`.

[35] J. Kwon, P. Varaiya, and A. Skabardonis. Estimation of truck traffic volume from single loop detector using lane-to-lane speed correlation. In *National Research Council (U.S.). Transportation Research Board. Meeting*, 2003.

[36] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks. In *OSDI*, 2002.

[37] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In *ACM SIGMOD*, 2003. To Appear.

[38] S. Madden, W. Hong, J. M. Hellerstein, and M. Franklin. TinyDB web page. `http://telegraph.cs.berkeley.edu/tinydb`.

[39] A. Mainwaring, J. Polastre, R. Szewczyk, and D. Culler. Wireless sensor networks for habitat monitoring. In *ACM Workshop on Sensor Networks and Applications*, 2002.

[41] NASA. Ceres info. Web page, September 2003. `http://eosdatainfo.gsfc.nasa.gov/eosdata/terra/ceres/ceres.html`.

[42] S. Oh, S. G. Ritchie, and C. Oh. An improved travel-time estimation algorithm using dual loop detectors. In *National Research Council (U.S.). Transportation Research Board. Meeting*, 2002.

[43] J. Rabaey, J. Ammer, T. Karalar, S. Li, B. Otis, M. Sheets, and T. Tuan. PicoRadios for Wireless Sensor Networks: The Next Challenge in Ultra-Low-Power Design. In *Proceedings of the International Solid-State Circuits Conference*, San Francisco, CA, Feburary 2002. Also see: `http://bwrc.eecs.berkeley.edu/Publications/2002/presentations/isscc2002/ISSCCslides.pdf`.

[44] S. Sastry, S. Schaffert, L. Schenato, S. Simic, B.Sinopoli, E. Brewer, D. Culler, and D. Wagner. Development of the NEST Challenge Application: Distributed Pursuit Evasion Games, 2002. Presentation. `webs.cs.berkeley.edu/weekly/Berkeley_challenge_app.pdf`.

[45] Sensoria Corporation. sGate wireless sensor gateway. Data Sheet. `http://www.sensoria.com/downloads/sGate%20Brochure%2003217-01%20Rev%20A.pdf`.

[47] SPSS, Inc. SPSS - Home. Web Page, 2003. `http://www.spss.com`.

[48] J. W. C. van Lint and N. J. van der Zjipp. An improved travel-time estimation algorithm using dual loop detectors. In *National Research Council (U.S.). Transportation Research Board. Meeting*, 2003.

[49] Washington State Department of Transportation. SR520 Ramp Metering at Montlake and Lake Washington Boulevards. Web page, 2002. `http://www.wsdot.wa.gov/Projects/SR520RampMeters/loopdetector.htm`.

[50] Y. Yao and J. Gehrke. The cougar approach to in-network query processing in sensor networks. In *SIGMOD Record*, September 2002.