

## OSOITE: TOWARDS REAL-WORLD SEARCH

Niwat Thepvilojanapong<sup>†,‡</sup> Masayuki Iwai<sup>†,‡</sup> Koichi Yamada<sup>†,‡</sup> Yoh Shiraiishi<sup>‡,‡</sup>  
Kazunori Takashio<sup>‡,‡</sup> Kazunori Umeda<sup>‡,‡</sup> Yosuke Tamura<sup>¶,‡</sup> Yoshito Tobe<sup>‡,‡</sup>  
<sup>†</sup>Tokyo Denki University, Japan <sup>‡</sup>University of Tokyo, Japan <sup>§</sup>Keio University, Japan  
<sup>‡</sup>Chuo University, Japan <sup>¶</sup>Fixstars Corp., Japan <sup>‡</sup>CREST, JST, Japan

E-mail: †{wat, tailor}@osoite.jp, yamada@im.dendai.ac.jp, yoshito\_tobe@osoite.jp

‡siraisi@csis.u-tokyo.ac.jp §kaz@mkg.sfc.keio.ac.jp ‡umeda@mech.chuo-u.ac.jp ¶tamura@fixstars.com

### ABSTRACT

This paper proposes a new architecture called *Overlay-network Search Oriented for Information about Town Events (OSOITE)*. OSOITE aims at providing users of personal devices with information about secure and safe actions in a usual urban life as well as in an emergency by utilizing the real-world overlay sensor networks. Specifically, we provide the users with alarms in a background and safe navigation based on the extracted high level information obtained with search in different types of database including real-time sensed data. Preliminary experiments using OSOITE prototype have been conducted, and the results are reported in the paper.

### KEY WORDS

Modelling, estimation, real-world search, sensor networks.

## 1 Introduction

Wireless sensor networks have received significant attention because sensing data observed from the physical world are useful for a number of emerging applications and our daily lives. A simple method to acquire sensing data is to request every sensor to report observed data periodically. Such process of collecting data in a sensor network has been verified through numerous testbeds and field experiments conducted by scientists [2]. Instead of waiting for periodic reports from sensor nodes, users, however, prefer querying the networks in a real-time fashion, i.e., the users submit a query whenever they have a question. Generally, people perform searching when they need an answer for something, so they are likely to search things related to their daily lives in the real world ranging from straightforward queries to complex ones. Examples of queries are weather forecast, an available parking lot, a list of nearby restaurants, route navigation, and so on. In order to expedite and realize the usage of sensor networks in a ubiquitous society, the sensor networks are expected to respond to such real-world queries. However, it is impossible that a sensor network has all kinds of information relevant to all queries, unless the queries are fixed or limited to specific types. It is also impractical to install all kinds of sensors (temperature, humidity, light, ...) in an area of interest. Even if the relevant kinds of information exist in the sensor

network or the queries are limited to some specific types, the sensor network needs to be deployed in a wide area (city or country level) so as to answer the queries correctly. Because deploying a variety of sensors throughout a city or country is impractical and inefficient, a collaboration of heterogeneous sensor networks is a possible and promising solution.

Currently, each organization deploys a sensor network for their own uses (e.g., [1, 7]), and sensing data are stored in central servers which can only be accessed by internal personnel. To share observed data with others, SensorMap [9], for example, tries to collect sensing data from heterogeneous sensor networks by allowing data owners to publish their data on the web. If published data are stored at central servers, the problems of bottleneck or high delay are likely to occur. Also large amount of bandwidth are required to collect data from numerous sensor networks. It wastes bandwidth if collected data are not useful for any searches. The lack of service infrastructures among heterogeneous sensor networks operating under different administrative organizations or agencies hampers the full use of real-world sensing data. In order to provide real-world search in heterogeneous sensor networks, we need an architecture that can provide relevant sensor data collected from large-scale heterogeneous sensor networks in a real-time fashion. In addition, current and past sensing data need to store in distributed manner amongst sensor networks, and a virtual database system of heterogeneous sensor network must be constructed to deal with real-world search according to users' requests.

In this paper, we propose a new architecture called *Overlay-network Search Oriented for Information about Town Events (OSOITE)*. OSOITE aims at providing users of personal devices with information about secure and safe actions in a usual urban life as well as in an emergency by utilizing the real-world overlay sensor networks. Specifically, we provide the users with alarms in a background and safe navigation based on the extracted high level information obtained with search in different types of database including real-time sensed data. Each sensor network works collaboratively by forwarding queries or searches to other networks that are likely to answer the queries. In addition, OSOITE is also designed to deal with multi-resolution queries, where resolution is a requirement determined by

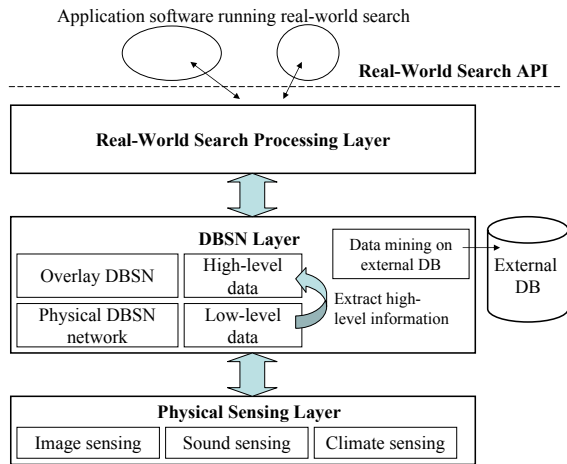


Figure 1. OSOITE architecture.

users. If sensing data collected from all networks do not meet the required resolutions, an interpolation is used to estimate results from coarse-grained data. The interpolation is also applied, if sensing data on requested area do not exist.

The remainder of the paper is organized as follows. Section 2 proposes OSOITE architecture. Section 3 describes modelling of real-world search. Section 4 presents prototype implementation and evaluates the OSOITE architecture through real-world experiments. Section 5 discusses related works. Section 6 concludes the paper.

## 2 OSOITE Architecture

OSOITE architecture (Figure 1) composes of the following main functions.

- Database querying process for real-world search.
- An API for building general application software.
- Extraction of high-level information from low-level data.
  - Networking of distributed database to form a virtual database.
  - Physical sensing infrastructure.

### 2.1 Real-World Search

Sensor information from heterogeneous sensor networks can be combined to indicate real-world condition such as user's status, changes of surrounding environment, and so on. Not only sensor information in the current state, but past information are also collected in database with the hopes of supporting people's daily life through real-world search. OSOITE offers real-world search by the means of searching for past, present, and future events in the real world, and providing results which is easy to understand to users. Future events can be estimated from past and/or present information. Before storing sensing data in

database, OSOITE inserts location information and time, and then performs high-level processing (data mining, etc.) for the purposes of real-world search. In addition, OSOITE also combines such sensed information with other kinds of information such as geographical information, statistic information, and web information in order to provide high-level information according to users' requests.

Current web searching technique is a querying process on static data, while real-world search is a process on dynamic data which change all the times both spatial and temporal context. OSOITE provides a network infrastructure for collecting sensing data from heterogeneous sensor networks, and a query processing engine for searching real-world information over heterogeneous overlay database. In addition, querying or searching model and description language are also developed as a part of search engine. An API is prepared as a tool to develop application software.

### 2.2 Application Framework for Real-World Search

We already know the benefit of storing past data by numerous Internet applications. Therefore, we strongly believe that storing real-world information and phenomenon and constructing searching infrastructure will help to support our daily lives. We need to provide real-world search as a stable service that works anytime and anywhere on any environment. In other words, users can use any terminals, which may have different processing resources or I/O interfaces. The service should also be available even if network resources (bandwidth, etc.) are different. OSOITE aims to prepare an application framework that fills up a gap between searching API and users as stated above. Based on such application framework, applications related to safety in daily life (e.g., safe navigation or alarms in a background mode) can be developed and provided as services to users.

### 2.3 Real-Time Extraction of Context

To realize safe society through real-world search, we need to know information related to safety exactly and in time in order to avoid, protect, or alleviate undesired events such as a natural disaster Tiny-sized wireless sensor nodes which are distributed in a space must detect primitive information, and extract related information in real time. Information is then aggregated and sent to external database for later use. High-level data in external database can be retrieved on-demand and real-time which are useful when disaster or any urgent events occur. In particular, OSOITE architecture employs a concept of approximation reasoning algorithm based on Bayesian network. We are developing a new probabilistic inference model to distributively extract high-level information from low-level data in real time.

### 2.4 Database Sensor Network

Database Sensor Network (DBSN) is a collection of database that spread over heterogeneous sensor networks.

Sensing data are not stored at a central database but they are distributed throughout multiple database in each sensor network. DBSN peer keeps high-level information and information created from different kinds of database. Because there are numerous kinds of sensing data and resolutions of the same data are also different, it is necessary to combine sensing information from multiple DBSN peers. Therefore, OSOITE architecture proposes to use a routing protocol based on attributes of data in an overlay network. Message exchange protocol between distributed DBSN peers is designed and will be released as DBSN software. Since DBSN peers may be switch between on or off frequently, the protocol is responsible to manage overlay routing dynamically.

## 2.5 Physical Sensing Layer

Image data is a powerful sensing data when considering providing safety using real-world search in sensor networks. We pay attention on movement information which are recognized as high-value data and can be used in various applications. We have developed a robust camera system using small-sized stereo cameras to detect, measure, and collect movement and distance information simultaneously both indoor and outdoor. Unlike existing stereo camera, each camera in our system extracts movement area, and then conducts stereo matching of such detected areas. As a result, we can realize stable measurement of distance, and can acquire both movement and distance information simultaneously. Because we also get real size of movement area, numerous kinds of query can be responded by this system. For example, human or cat entered the room, or how many people are in the area. Our final objective is to use 15-centimeter portable camera which can work as a stand-alone device without connecting to a PC or laptop computer. We can use movement and distance information with other kinds of sensing information (sound, climate, etc.) in order to provide safety exactly and in time.

## 3 Modelling of Real-World Search

Since there are many kinds of sensing data, and the same sensing data also have different resolution, we need to construct a model for real-world search. First, we describe a basic model for real-world search, then we propose a method to deal with multi-resolution search.

### 3.1 Basic Search Model

The current prototype of OSOITE architecture uses a concept of *LTE-tuple* ( $\Omega$ ) to process real-world searches. The *LTE-tuple* consists of three components: location ( $L$ ), time ( $T$ ), and event ( $E$ ), which are the input of the API.

$$\Omega = (L, T, E) \quad (1)$$

Each component of the tuple can be specified as follows.

$$L := \{ \text{point(s)} \mid \text{line segment(s)} \mid \text{closed region(s)} \}$$

$$T := \{ \text{point(s) of time} \mid \text{period(s)} \}$$

$$E := \{ \text{raw phenomenon or conditions} \mid \text{deduced phenomenon or conditions} \}$$

These three components are search conditions specified by the users. The users input only component(s) they desire, then the left component(s) will be searched by the search engine. There are seven patterns of this search model.

(1) Specify  $L$ , and search for  $T$  and  $E$ . For example,  $L$  = a point in Shinjuku area. The output  $(T, E) = (17:00, \text{the quarrel}), (18:00, \text{the car accident}), \text{etc.}$

(2) Specify  $T$ , and search for  $L$  and  $E$ . This search is possible but it is nonsense.

(3) Specify  $E$ , and search for  $L$  and  $T$ . For example,  $E$  = temperature  $\geq 30^\circ\text{C}$ . The output  $(L, T) = (\text{Kanda intersection}, (11:45, 15:20)), (\text{in front of Kanda police box}, (11:50, 15:10)), \text{etc.}$

(4) Specify  $L, T$ , and search for  $E$ . For example,  $(L, T) = (\text{Kanda shopping street}, [13:00, 15:00])$ . The output  $(L, T, E) = (\text{in front of Kanda Foreign Language School}, 13:30, \text{a guy fell down due to heat wave}), \text{etc.}$

(5) Specify  $L, E$ , and search for  $T$ . For example,  $(L, E) = (\text{a point in Hachioji area}, \text{an unlawful dumping of garbage})$ . The output  $T = 1:23, 3:50, 23:10$ .

(6) Specify  $T, E$ , and search for  $L$ . For example,  $(T, E) = ([18:00, 22:00], \text{purse-snatching})$ . The output  $L = \text{in front of Ueno train station}$ .

(7) Specify  $L, T, E$ , and narrow the search. For example,  $(L, T, E) = (\text{the area from Kanda station to Tokyo Denki University}, \text{current time}, \text{walking through a breezy path})$ . The output  $L = \text{a specific path from Kanda station to Tokyo Denki University}$ .

The DBSN peer processes the searches and creates responses according to the above *LTE-tuple*'s principle. This search model can be extended by including other components in the tuple according to the requirement of application software by using OSOITE API.

### 3.2 Multi-Resolution Search

This section describes how a network of DBSN peers work collaboratively to response multi-resolution searches. Without loss of generality, we discuss only spatial resolution here as an example. Other kinds of resolution (e.g., temporal resolution) can be performed in the same way. The DBSN peers have limited sensing data because size of each sensor network is limited. Based on the collected data, the DBSN peers know spatial resolution of sensing data in their region. To satisfy the required resolutions, each DBSN peer keeps the following information in its *peer information table*: DBSN peer ID, data type (temperature, light, etc.), region, and spatial resolution ( $R_s$ ). The DBSN peers always exchange the table with other DBSN peers using overlay routing protocol.

When receiving a query, a DBSN peer checks whether its data satisfy the query's requirements which compose of

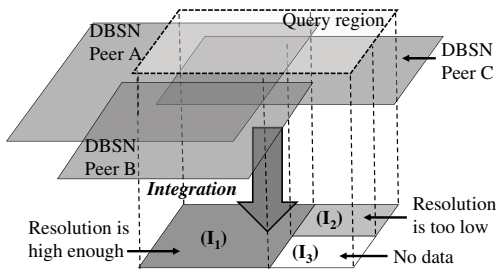


Figure 2. Sensing data from three DBSN peers are integrated as shown in the bottom plane.

region and spatial resolution. Only DBSN peers that satisfy both requirements send responses to user. The requirement on region is satisfied, if any part of DBSN peer's region overlaps the query's region. The spatial resolution is satisfied, if DBSN peer's resolution is finer than that specified in the query. If the requirements are not satisfied, the DBSN peers forward the query to other DBSN peers that are likely to respond to this query. Forwarding decision is based on the peer information table exchanged with other DBSN peers.

According to the above procedures, user may get multiple responses because regions of multiple DBSN peers overlap the query's region. Integration and interpolation of multiple responses are performed to acquire more accurate result. Figure 2 is an example showing three DBSN peers whose regions overlap the query region (the upmost plane). The bottom plane in the figure is a result of integrating data from those three DBSN peers. There are three possibilities of region integration. First, the resolution of overlapped region is equal to or finer than the required resolution ( $I_1$ ). Second, the resolution of overlapped region is coarser than the required resolution ( $I_2$ ). Third, some parts of query region does not overlap region of any DBSN peers ( $I_3$ ). In the case of  $I_1$ , the response can be created from received data. However, the *Inverse-Distance Weighting (IDW)* [10] is employed as an interpolation technique for the case  $I_2$  and  $I_3$ . The responses are created according to the above LTE-tuple's principle.

## 4 Prototype Implementation and Experiments

We developed a prototype of OSOITE and performed preliminary experiments. The prototype implementation consists of a DBSN requester, three DBSN peers, and sensor nodes. The DBSN requester receives searching queries from users. The DBSN requester and DBSN peers are Linux-based machines implemented with C language. The DBSN sources or sensor nodes are the uParts [12], which are deployed in the nearby area of the associated DBSN peers. The USB bridges [12] are attached to the DBSN peers in order to collect temperature data measured by the

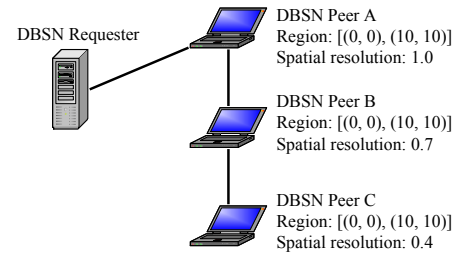


Figure 3. Network topology and attributes of each DBSN peer in Experiment I.

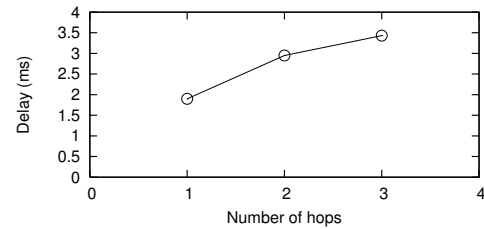


Figure 4. Delay measured in Experiment I.

nearby uParts. We use the PostgreSQL as a database management system (DBMS) in our implementation. For the sake of simplicity and privacy concern, we use relative coordinates to specify rectangular regions in the experiments. The regions are expressed by lower-left and upper-right coordinates:  $[(x_{ll}, y_{ll}), (x_{ur}, y_{ur})]$ . The spatial resolution used in the experiments is a distance between two sensor nodes so that lower value means finer data (or finer resolution).

To study the performance of OSOITE prototype in the real-world environment, we carried out two experiments and measured delays of responses. The delay is defined as a period between transmitting a search and receiving the last response related to the search at the DBSN requester.

### 4.1 Experiment I: Resolution-based Queries

The first experiment is prepared as illustrated in Figure 3. The DBSN requester always injects queries to DBSN peer A according to the experiment setup. The region attribute of three DBSN peers (A, B, and C) are the same, i.e., a square region  $[(0, 0), (10, 10)]$ . The number of sensor nodes associated with each DBSN peers is varied so that the spatial resolutions of DBSN peer A, B, and C are 1.0, 0.7, and 0.4, respectively. The DBSN peer C has the finest resolution in this scenario. Note that the region is fixed so we vary the resolutions by changing the number of sensor nodes.

The DBSN requester injected three kinds of query by varying only spatial resolution, i.e., other fields of the queries including event ( $E =$  all measured temperature),

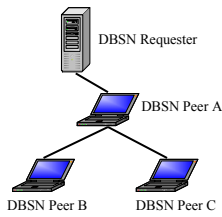


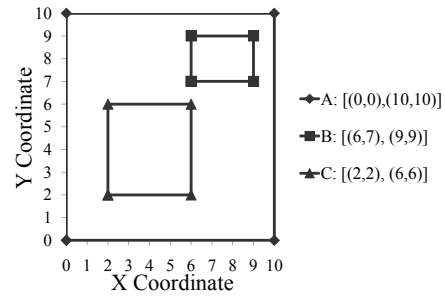
Figure 5. Network topology used in Experiment II.

region ( $L = [(0,0), (10,10)]$ ), the period of time ( $T = [18:50, 18:55]$ ), and temporal resolution are the same. The spatial resolutions of queries are set to 1.0, 0.7, and 0.4, thereby the queries traverse one, two, and three hops, respectively. The details of query processing are as follows. If the spatial resolution of query is set to 1.0, only DBSN peer A responds to the query. If the spatial resolution of query is set to 0.7, DBSN peers A forwards the query to DBSN peer B because A's data does not satisfy the required resolution. DBSN peer B sends the response to the DBSN requester; however, it does not forward the query further because its data satisfy the required resolution. If the spatial resolution of query is set to 0.4, DBSN peer A forwards the query to DBSN peer B which in turn forwards the query to DBSN peer C because data stored in DBSN peers A and B does not satisfy the required resolution. DBSN peer C then sends the response to the DBSN requester. The delays of three queries are shown in Figure 4, where the number of hops in the horizontal axis implicitly represents the resolutions specified by each query. In particular, one, two, and three hops mean resolutions 1.0, 0.7, and 0.4, respectively. The number of hops makes the graph meaningful because the delay increases as the number of hops increases. This experiment shows that finer resolution requires longer delay of response. We conclude that multi-resolution queries are handled by collaborative works of heterogeneous sensor networks. OSOITE architecture and the prototype implementation respond correctly when users ask for different resolutions of information.

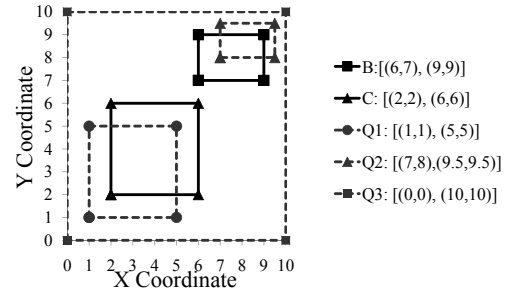
#### 4.2 Experiment II: Region-based Queries

Figure 5 illustrates network topology of Experiment II in which queries ask for temperature data of different regions. The region attributes of DBSN peers A, B, and C are  $[(0,0), (10,10)]$ ,  $[(6,7), (9,9)]$ , and  $[(2,2), (6,6)]$ , respectively. The graphical view of region attributes is depicted in Figure 6(a). The DBSN requester also injects three queries in this experiment. Figure 6(b) delineates the regions specified by three queries (Q1, Q2, and Q3) which are  $[(1,1), (5,5)]$ ,  $[(7,8), (9.5,9.5)]$ , and  $[(0,0), (10,10)]$ , respectively.

Query processing are as follows. Because DBSN peers A and C hold temperature data of the region requested by Q1, DBSN peer A sends the response to the



(a) Region attributes of each DBSN peer.



(b) Regions specified by three queries (Q1, Q2, and Q3). Region attributes of DBSN peers B and C are also shown in the figure.

Figure 6. Regions of DBSN peers and queries.

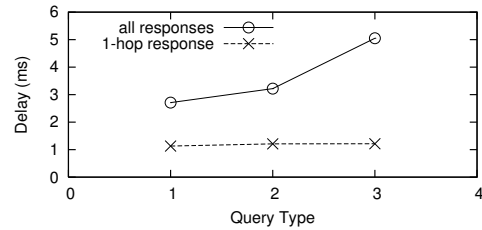


Figure 7. Delay measured in Experiment II.

DBSN requester and forwards the query Q1 to DBSN peer C. DBSN peer C then sends the response to the DBSN requester. The region specified by Q2 overlaps the regions associated with DBSN peers A and B. After receiving the queries, DBSN peers A and B send the responses to the DBSN requester. Because the region of Q3 overlaps those of three DBSN peers, all DBSN peers send the responses to the DBSN requester. Figure 7 shows the delays of one-hop response and all responses. The one-hop delay is a period between transmitting a query and receiving the response from DBSN peer A which stays one hop from the DBSN requester. The all-response delay is the delay we defined above, i.e., a period between transmitting a query and receiving the last response related to the query at the DBSN requester. The one-hop delay is quite comparable for all of three queries because DBSN peer A sends the response before forwarding the query to other DBSN peers.

The all-response delay of Q3 is higher than those of Q1 and Q2 because three responses need by Q3 while only two responses need by Q1 and Q2. The all-response delay of Q2 is slightly higher than that of Q1 because of physical factors such as CPU speed, Ethernet speed, and so on. This experiment shows that OSOITE prototype also works with region-based queries.

## 5 Related Work

There are many fields of literature related to our work. TinyDB [8] which operates on TinyOS [13] is unaware of the existence of sensor network. Sensing data is aggregated and searched through tree structure. Shimizu et al. [11] develop CRUISE/r, a system to search for lost item in the real world. Frank et al. [4] extend CRUISE/r by showing lost items in the Internet. Unlike both TinyDB and CRUISE/r, OSOITE architecture can be used in wide range of applications because it consists of many components that have flexible uses. Moreover, OSOITE pays attention on real-world search related to safety of daily life.

The notion of multi-resolution has been explored in many ways. Ganesan et al. [5] proposed a multi-resolution storage system that generates multi-resolution data from raw sensing data by using wavelet decomposition. Their system is designed as a single database system because they assumed a single sensor network. However, we assume heterogeneous sensor networks that are widely distributed in different regions and have different resolutions of sensor data. We aim at connecting heterogeneous database repository by allowing each repository to announce its resolution, thereby eliminating the necessity of wavelet decomposition.

For in-network query aggregation, several methods [6, 8] have been proposed under the assumption of limited resources such as battery lifetime and communication bandwidth. In addition to simple operations including union, intersection, and extraction of the maximum, the minimum, or the mean value, more statistics-dependent schemes based on the distribution of sensor data are proposed. They are common in that aggregation is performed at a sensor node. In contrast to these works, our scheme performs an interpolation or aggregation operation to an area which may or may not contain sensor nodes.

As a database system for sensor data, MauveDB [3] has been proposed. MauveDB considers uncertainty of sensor data and creates meshed data utilizing regression and interpolation with the notion of view. Depending on a view, a user cannot see original data including uncertainty. Our system differs from MauveDB in that distributed connection of database is used.

## 6 Conclusion and Future Works

Real-world search is a promising application for sensor networks. OSOITE is architecture for realizing such

real-world search through heterogeneous sensor networks. Those sensor networks construct an overlay network to work collaboratively. Sensing data with different characteristics are stored locally in each network. An API is prepared for developing application software running real-world search. A Search or query is sent through the API into the network. Then the query is forwarded to the node whose stored data matches the conditions specified in the query. If sensing data are not available or resolution is not fine enough, the interpolation technique is used to estimate results. We have implemented OSOITE in the real-world environment and performed the experiments to study its performance. The experiments showed that OSOITE architecture responds correctly when users ask for different resolutions or regions. Queries are forwarded to related DBSN peers through the Internet, and delay increases as the number of hops increases. We plan to perform large-scale experiments and include other sensing information in the future.

## References

- [1] J. Burke et al. Participatory sensing. In *Proceedings of the Workshop on World-Sensor-Web (WSW'06)*, Nov. 2006.
- [2] Alberto Cerpa et al. Habitat monitoring: application driver for wireless communications technology. *SIGCOMM Computer Communication Review*, 31(2 supplement):20–41, 2001.
- [3] Amol Deshpande and Samuel Madden. MauveDB: Supporting model-based user views in database systems. *ACM SIGMOD*, pages 373–384, June 2006.
- [4] C. Frank, C. Roduner, and C. Noda. Query scoping for the sensor internet. *ICPS'06*, June 2006.
- [5] Deepak Ganesan et al. Govindan. Multiresolution storage and search in sensor networks. *ACM Transactions on Storage*, 1(3):277–315, August 2005.
- [6] Johannes Gehrke and Samuel Madden. Query processing in sensor networks. *IEEE Pervasive Computing*, 3(1):46–55, Jan.–March 2004.
- [7] Hock Beng Lim et al. The national weather sensor grid. *SenSys'07*, pages 369–370, November 2007.
- [8] Samuel R. Madden et al. TinyDB: an acquisitional query processing system for sensor networks. *ACM Transactions on Database Systems (TODS)*, 30(1):122–173, March 2005.
- [9] Suman Nath et al. SensorMap: A web site for sensors worldwide. *SenSys'06*, pages 373–374, November 2006.
- [10] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 23rd ACM National Conference*, pages 517–524, August 1968.
- [11] Hirobumi Shimizu et al. Association management between everyday objects and personal devices for passengers in urban areas. *Pervasive 2005, Demonstrations*, pages 89–92, May 2005.
- [12] TecO/University of Karlsruhe. Particle Computer. available from <http://particle.teco.edu/>.
- [13] TinyOS. available from <http://www.tinyos.net/>.