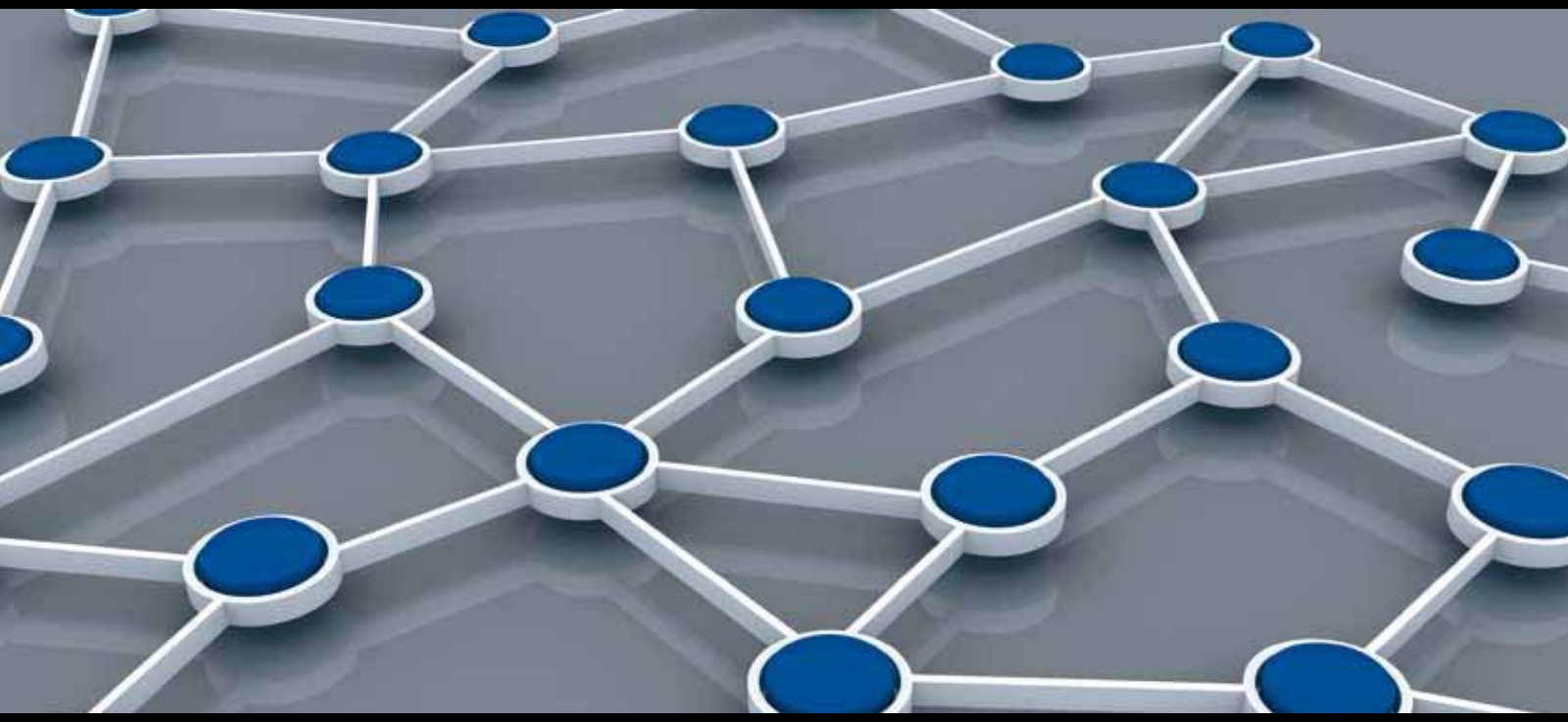
International Journal of Distributed Sensor Networks

Cyber Physical Systems Using Sensor Technologies

Guest Editors: Chih-Yung Chang, Sherali Zeadally, and Tzung-Shi Chen



Cyber Physical Systems Using Sensor Technologies

International Journal of Distributed Sensor Networks

Cyber Physical Systems Using Sensor Technologies

Guest Editors: Chih-Yung Chang, Sherali Zeadally, and Tzung-Shi Chen

Editorial Board

Contents

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 957396, 2 pages
doi:10.1155/2012/957396

Editorial

Cyber Physical Systems Using Sensor Technologies

Chih-Yung Chang,[1] Sherali Zeadally,[2] and Tzung-Shi Chen[3]

[1] *Department of Computer Science and Information Engineering, Tamkang University, New Taipei 25137, Taiwan*
[2] *Department of Computer Science and Information Technology, University of the District of Columbia, Washington, DC 20008, USA*
[3] *Department of Computer Science and Information Engineering, National University of Tainan, Tainan 700, Taiwan*

Correspondence should be addressed to Chih-Yung Chang, cychang@mail.tku.edu.tw

Received 6 September 2012; Accepted 6 September 2012

Cyber-physical systems (CPSs) have grown exponentially and have been attracting a lot of attention over the last few years because of cyber and the physical (natural and human-made) components that are being tightly integrated into many systems. Smart or intelligent sensor networks should be provided with the 3C (computation, communication, and control) abilities. Emerging applications include transportation, energy, healthcare, manufacturing, entertainment, consumer electronics, environment monitoring, and many parts of our social infrastructure. There are eight papers selected for publication in this special issue. Most of the selected papers present the design and implementation of cyber-physical systems applied to several areas, including safety surveillances, rehabilitation, smart living space, and so on.

G. Manes et al. propose a continuous remote monitoring platform which consists of sensor nodes and a gateway. The sensor nodes deployed at the hazardous sites will forward data to the platform through the gateway to improve the transmission reliability. The proposed platform uses volatile organic compound (VOC) detectors which are able to perform real-time analysis in potentially hazardous sites at unprecedented time/space scale. Overall, the platform provides an easily deployable stand-alone infrastructure with a high degree of scalability and reconfigurability with minimal intrusiveness or obtrusiveness.

C.-M. Own proposes the pet care system by establishing a location-aware infrastructure such that pet owners can timely monitor the pet's situation. The system applies the concepts of CPS and provides pet owners with the ability to remotely obtain their pet information and control the smart home of the pet. To monitor the behavior of the pet, a sensor tag is fixed on the collar of the pet and a location-aware algorithm was proposed to improve the location accuracy of the pet. Based on the pet's behavior, an intelligent pet door and a smart pet feeder are implemented using smart-home technology. Consequently, the proposed system can meet the needs of many pet owners.

One important topic in CPS research is how to retrieve the events from massive amounts of sensor data and analyzing them with spatial, temporal, and other multidimensional information. L.-A. Tang et al. propose the concept of a typical cluster which is a model describing multidimensional features of a typical event. The typical clusters can be efficiently integrated into a hierarchical framework to form macroclusters for large-scale analytical queries. To retrieve significant macroclusters, the system employs a guided clustering algorithm to filter out the trivial results and thus guarantees the accuracy of significant clusters. The proposed mechanisms are evaluated on gigabyte-scale datasets from real applications and save considerable time cost of the baselines.

C.-Y. Lin et al. present the most important design requirements of CPS architectures and summarize key sensor network characteristics that can be leveraged in CPS designs. The authors review many well-known CPS application domains that depend on wireless sensor networks (WSNs) in their design architectures and implementations.

The challenges that still need to be addressed to enable seamless integration of WSN with CPS designs are also discussed.

J.-F. Chen et al. propose the pelvic floor muscle training (PFMT) system which is aimed help patients execute their rehabilitation exercises. The proposed PFMT system consists of the PFMT devices, an Arduino control board, a force sensor, a Bluetooth device, and the secure digital memory (SD) card. The embedded force sensor automatically monitors the operations of patients and the Bluetooth device sends timely reports if the PFMT exercise is done incorrectly. By applying the proposed PFMT system, the doctors automatically obtain the behavior in executing pelvic floor muscle training and contact patients for additional visitation(s) if necessary.

Based on CPS technology, creating smart living space becomes an important trend of future development. However, one challenge encountered in establishing a smart living space is that the electronic devices nowadays execute different communication protocols, such as Bluetooth, Zigbee, RF, infrared, among others and even some traditional devices have no communication functions. Z.-Y. Bai and X. Huang design and implement an intelligent control box to convert different wireless signals. The developed intelligent control box can be treated as a multiple control platform which integrates the systems of lighting, air conditioning, access control, video surveillance, alarm, and so on. The proposed intelligent control box provides several control functionalities, including the systems of lighting, air conditioning, access control, video surveillance, alarm, and so on. The proposed intelligent control box automatically converts different wireless signals and removes the difficulties in establishing smart living space with CPS technology.

M. B. Shah et al. propose a context-aware routing protocol for multisink Cell-Phone-based sensor networks (CPSNs). Since the data delivery to the sink is delay tolerant, this paper exploits the mobility of the cell phone (human mobility) to opportunistically forward data to the sink. This approach can be especially applied to the CPSN which is characterized by volatile topology, limited buffer space, and loose connectivity with neighboring nodes. The authors propose a human mobility sensor context aware routing (HMSCAR) protocol-based on human mobility patterns aimed at predicting the best relays for successfully forwarding data to the sinks.

Chalermek Intanagonwiwat proposes the declarative resource naming (DRN) framework to simplify the programming of wireless networks of embedded systems (WNESs). The nodes of WNES can be deployed in dynamic and hostile environments even though the operator may not be able to physically reach them. The proposed DRN framework abstracts low-level details in system and network programming, and it provides many good features including programming simplicity, expressiveness, tunability, on-the-fly reprogrammability, and in-network data aggregation for energy savings. The proposed DRN framework has been implemented on two platforms: Smart Message and Maté, to verify that DRN enables programmers to develop energy-efficient applications with the desired flexibility and quality.

We believe that, in all cases, the selected papers are definitely important ones for this special issue on cyber physical systems using sensor technologies. The editors would like to thank all the authors who have submitted their manuscripts for consideration for this issue. We thank all reviewers for their efforts in reviewing all submitted papers which greatly helped us select the finest papers for this issue.

Chih-Yung Chang
Sherali Zeadally
Tzung-Shi Chen

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 490826, 15 pages
doi:10.1155/2012/490826

Research Article

Hybrid Macroprogramming Wireless Networks of Embedded Systems with Declarative Naming

Chalermek Intanagonwiwat

Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok 10330, Thailand

Correspondence should be addressed to Chalermek Intanagonwiwat, chalermek.i@chula.ac.th

Received 16 February 2012; Accepted 11 May 2012

Academic Editor: Sherali Zeadally

Wireless Networks of Embedded Systems (WNES) are notoriously difficult and tedious to program. The difficulty is mostly originated from low-level details in system and network programming. This includes distributedly managing and accessing resources from a dynamic set of nodes in hostile and volatile networks. To simplify WNES programming, we propose *Declarative Resource Naming* (DRN) that abstracts out the mentioned low-level details by programming a WNES in the large (*i.e., macroprogramming*). DRN provides programming simplicity, expressiveness, tunability, on-the-fly reprogrammability, and in-network data aggregation for energy savings. None of existing macroprogramming paradigms supports all of the mentioned features. Furthermore, DRN is an integration of declarative and imperative programming. The low-level details are declaratively abstracted out, but the main algorithm remains procedural. This allows programming simplicity without an adverse impact on the expressiveness. We have implemented and evaluated DRN on two platforms: Smart Message and Maté. Our result indicates that DRN enables programmers to develop energy-efficient applications with the desired flexibility and quality.

1. Introduction

A WNES (e.g., a wireless sensor network) consists of a massive number of resource-constrained wireless nodes that are unattendedly deployed to collect data or to monitor the area in dynamic, hostile environments. Programming WNES for such applications is notoriously difficult and tedious because of the low-level details in system and network programming. These low-level details include distributedly discovering, managing, and accessing remote resources as well as routing in a dynamic set of nodes while maintaining low energy consumption and memory usage.

Several programming abstractions have been proposed in literature to hide these low-level details from the programmers. Of particular interest are approaches to program WNES in the large (i.e., macroprogramming). Unlike other abstractions, these macroprogramming abstractions allow programmers to take a centralized view of programming a distributed system rather than a distributed view. A macrocompiler is normally required for translating a centralized-view macroprogram into a distributed version for execution. The macrocompiler is also responsible for automatically generating the mentioned low-level details in the executable distributed code.

Macroprogramming abstractions can be divided into two subclasses: node-independent and node-dependent. In node-independent subclasses, WNES is declaratively programmed as a whole or a unit (e.g., a database). Examples of node-independent abstractions are TinyDB [1], Cougar [2, 3], and Sense2P [4, 5]. By abstracting a WNES as a database, WNES programming is reduced to database querying.

Conversely, in node-dependent subclasses, WNES are programmed as a collection of nodes. These abstractions enable programming tasks that are more complicated than database-like querying. Examples of node-dependent abstractions include Kairos [6], Split-C [7], SP [8, 9], Regiment [10], Macrolab [11], and EcoCast [12]. Most of these works (except Regiment and Macrolab) do not support in-network data aggregation for energy savings. Even though Regiment and MacroLab do, they do not address the on-the-fly reprogrammability issue.

In this paper, we propose Declarative Resource Naming (DRN) (an initial design of this work appears in Algosensors [13]), a hybrid macroprogramming approach that supports simple tasks (e.g., database-like querying), and difficult tasks as well as data aggregation mechanisms for energy savings. DRN is an integration between declarative and imperative programming. The low-level details are declaratively abstracted out whereas the core algorithm remains procedural. Our abstraction allows programmers to declaratively describe a dynamic set of nodes by their run-time properties and to map this set to a variable. To access the desired resources on nodes in the set, we can simply refer to the mapped variable. Therefore, remote resource access is simplified to only variable access that is completely network-transparent. DRN provides both sequential and parallel access to the desired set. Parallel access reduces the total access time and energy consumption because it enables data aggregation in the network. Additionally, we can associate each set with tuning parameters (e.g., timeout, energy budget) to bound access time or to tune resource consumption.

Given that WNES may be deployed in dynamic, hostile environments, and also that we may not be able to physically reach the nodes, it is necessary that we can remotely program these unattended nodes on the fly. Systems based on code migration are preferable because programs can be propagated to target nodes without human intervention or system rebooting. Examples of such systems include Smart Messages (SM) [14], SensorWare [15], and Maté [16]. Therefore, we have implemented our DRN run-time library on two mobile-agent platforms: SM and Maté. SM can run on iPAQs equipped with 802.11 radios whereas Maté can run on motes equipped with 802.15.4 radios.

In addition, we have implemented an object tracking application using our DRN runtime library to illustrate the model's viability. We have also evaluated our DRN runtime library and its tuning knob (i.e., resource binding lifetime). Our result indicates that the tuning knob enables the DRN application to save up to 55.2% of the bytes sent without significant accuracy degradation.

2. Related Work

WNES macroprogramming has been explored earlier by several research efforts, including TinyDB [1], COUGAR [2, 3], Semantic-Streams [17], and Sense2P [4, 5, 18]. The above node-independent abstractions propose programming WNES as a database. Thus, WNES programming is reduced to database-like querying with declarative languages. However, declarative languages are designed for expressing the desired data, but not for expressing the algorithmic details. As a result, they are not appropriate for complex tasks where the core algorithmic details cannot be automatically generated. Conversely, DRN is a hybrid between declarative and imperative languages. Thus, our work can easily support both types of tasks.

Other macroprogramming research efforts are node-dependent. These include Kairos [6], SP [8, 9], Regiment [10], Macrolab [11], and EcoCast [12]. Similar to Split-C [7] for parallel programming, Kairos provides a facility to sequentially access remote variables for WNES programming. Unlike Split-C and Kairos, DRN can access variables and other resources at declaratively-named nodes in parallel. Accessing resources in parallel significantly reduces the total access time and the overall energy consumption (by enabling data aggregation inside the network).

Our work is mostly influenced by Spatial Programming (SP). DRN and SP simplify resource access as variable access, exposing the space property to the programmers, hiding network details, and supporting imperative programming. However, SP supports only sequential resource access, whereas DRN supports both sequential and parallel access. Additionally, SP is purely imperative programming, but DRN is partially declarative and mostly imperative.

Parallel access is also supported by other works such as EcoCast, Regiment, and Macrolab. Surprisingly, even though EcoCast does access resources in parallel, it does not support in-network data aggregation.

Regiment is a spatiotemporal macroprogramming system based on functional reactive programming paradigms (one form of declarative programming). In Regiment, the whole program can be treated like a math equation that reacts to the input changes. Input is data from a set of nodes that are defined by their location. In this sense, Regiment is very similar to SP. However, Regiment is not welldesigned for applications with highly dynamic behaviors, nonreactive applications, short-lived queries, or mobile-agent-based applications. Conversely, DRN does not suffer from the above limitations.

Macrolab is a Matlab-like macroprogramming framework that provides deployment-specific code decomposition. In Macrolab, every deployment change requires recompilation and reinstallation. This can be troublesome as there is no explicit support for remote-reprogramming the system on the fly. In contrast, DRN is incorporated with two mobile-agent platforms. Thus, our work can certainly handle such changes with ease.

There exist several research efforts on a hybrid of declarative and imperative programming. Examples of such research include embedded SQL [19] and constraint-imperative programming [20]. In embedded SQL, SQL is mainly used for database access, and imperative programming is used for data processing. In a sense, resources in DRN are analogous to the database in embedded SQL where declarative accesses are appropriate. In constraint-imperative programming, variables are confined with conditions about their eligible value. Given that conditions are declaratively described, our resource variables are similar to their constrained variables. Despite the mentioned similarity, DRN, embedded SQL, and constraint imperative programming target different problems, platforms, and environments. Specifically, embedded SQL is designed for data processing on conventional databases, and constraint-imperative programming is designed for computing a solution that matches a particular constraint on traditional systems. In contrast, DRN targets resource naming on highly dynamic WNES.

Table 1: WNES macroprogramming system characteristics.

System	Characteristic				
	Programming model	Node dependency	Supported tasks	In-network data aggregation	On-the-fly reprogramming
Cougar	Declarative (SQL)	Node independent	Relational database queries	Yes	No
TinyDB	Declarative (SQL)	Node independent	Relational database queries	Yes	No
Semantic Streams	Declarative (logic programming)	Node independent	Service queries	No	No
Sense2P	Declarative and imperative (logic programming)	Node independent	Deductive database queries	No	Yes
SP	Imperative (procedural programming)	Node dependent	Space-centric	No	Yes
Kairos	Imperative (procedural programming)	Node dependent	Remote variable access	No	No
EcoCast	Imperative (object-oriented programming)	Node dependent	Interactive group access	No	Yes
Regiment	Declarative (functional programming)	Node dependent	Spatiotemporal	Yes	No
Macrolab	Imperative (Matlab-like)	Node dependent	Deployment specific	Yes	No
DRN	Declarative and imperative (procedural programming with declarative names)	Node dependent	Declarative resource access	Yes	Yes

Hybrid macroprogramming systems also exist in Internet (e.g., XTree [21]). Similar to TinyDB, X-Tree programs the whole system as a database but X-Tree is designed for wide-area sensor systems, not hostile dynamic WNES.

Nevertheless, our work has been influenced by directed diffusion [22–24] and LEACH [25]. This is seen most clearly in the energy savings gained by processing data in the network. Despite this influence on our parallel access, DRN shares several similarities with diffusion. Given that diffusion APIs [26] require declarative data description for publication and subscription, DRN and diffusion are examples of hybrid programming that effectively hides networking details. However, diffusion programming view can be somewhat distributed. This is probably why diffusion is not widely classified as macroprogramming in the community.

We summarize the differences of these macroprogramming approaches in Table 1.

3. What Is the Right Abstraction?

Traditionally, there are two programming styles in computer literature: declarative and imperative. Declarative programming fully abstracts out all algorithmic details. Programmers only specify what they want rather than how to algorithmically obtain the results. The translator and optimizer will then fill in the algorithms. Automatic generation of algorithmic details can be efficient for simple and specific tasks (e.g., database), but is questionable for others. Examples of such an SQL-based approach include COUGAR and TinyDB. Despite its simplicity, declarative programming is not applicable for every WNES application. Imperative programming is more appropriate for complex tasks where efficient algorithmic details are either not obvious, or not easy to generate automatically. For example, it is difficult or even impossible to implement Kalman filters or maximum likelihood algorithms for estimating object locations in SQL because SQL is not designed for expressing algorithmic details.

Declarative and imperative programming function well within their domain and complement one another. Integration of declarative constraints and imperative constructs can form a powerful programming paradigm suitable for both domains. In this paper, we propose that such integration is possible if the declarative abstraction is applied only to some parts of the program.

In general, potential targets for abstraction are (1) parts that are unrelated to the core algorithms (2) common to applications, and (3) tedious for programmers. To identify the abstractable parts, a basic understanding of WNES programs is required. Typically, programs are collections of operations on variables and resources. Given that variables are more frequently accessed, programming languages provide a simpler way to access variables than to access resources.

Not surprisingly, traditional resource access is more tedious, especially in networked systems where there exists a distinction between local and remote resources. Resources are normally bound to nodes that are known a priori. Therefore, in order to specify the remote resources that are of interest, node ids are required. If the node ids are not known, resource discovery is needed. As a result, programmers are required to work on several programming details (e.g., networking, resource discovering, resource accessing).

WNES programming is even more labor-intensive because the resources of interest are specified by their properties at run-time rather than node ids. For example, we may want to access sensors on a particular hill only when the temperature is more than 30 degrees Celsius. In this

case, resource discovery in WNES becomes necessary and common rather than optional. The resource property is highly dynamic because the environment—where the temperature can drop below 30 degrees Celsius at any moment—is hostile and volatile. Some resource bindings or mappings may have to be invalidated because the bound resources may no longer match the desired property. But even if the resource property does not change, bound resources may not be accessible because of network dynamics such as node mobility. WNES programs are required to handle changes, invalidate bindings, discover equivalent resources, and bind the newly discovered resources. Given that the above events are frequent in WNES, these resource handlings (e.g., discovering, accessing, rebinding, and networking) are tedious to programmers. Therefore, the resource-related parts of the WNES program are reasonable choices for our declarative abstraction.

4. Declarative Resource Naming

To simplify the programming tasks for WNES, we propose a scheme that will program the WNES as a collection of nodes in a network-transparent manner. As a result, there is no notion of networking, being remote, or local.

4.1. Resource Variable. WNES programming can be simplified by making a resource access as simple as a variable access. In order to do this, we propose *resource variables* (i.e., variables that are mapped and referred to actual nodes). For example, one can write a program to read a light sensor and to control a camera as follows:

```
Resource R, X;
Printf "(light intensity = %f", R⟶ light);
X ⟶ camera = off;
```

In the above example, we assume that the resource variable R contains a light sensor and the resource variable X contains a camera. To read the light intensity, we can simply refer to $R \rightarrow light$. Similarly, the camera can be turned off by assigning off to $X \rightarrow camera$. There is no need for algorithmic detail of resource controls and operations. This example shows that our approach is not only for retrieving data and for pushing data to desired nodes but also for controlling them.

4.2. Declarative Constraint. Understandably, one may wonder to which physical nodes (or resources) these variables R and X) are precisely bound and how programmers know about the individual sensor types. Rather than specifying the node ids for binding, a target resource's desired property can be declaratively indicated with a boolean expression or a predicate. For example, we can specify that R will be bound to light-sensor nodes within the forest with temperatures greater than 30 degrees Celsius.

```
Resource R = ⟨within (location, forest) &&
                temperature >30 &&
                exist(light)⟩
Resource X = ⟨a(b,c)! = 0 && exist (camera)⟩
```

Given that more than one node can match a specified expression, a resource variable is referred to as a set of matching nodes rather than a single one. Location and temperature are local properties (of a node) that are used to determine the node's membership in the set R. Furthermore, we also allow user-defined boolean functions (e.g., function $a()$) in our expression. Such a flexible expression is generally powerful and sufficient for various complex conditions.

4.3. Resource Access. In this section, we illustrate the need for various types of DRN resource access that can be used in different situations. Their advantages and disadvantages are also provided as a guideline for selecting the resource access type that is most suitable for a particular task. We propose two approaches for accessing multiple matching nodes: *sequential* and *parallel*.

Sequential Access. Each element in a set can be referred to using an iterator (similar to an iterator in C++ standard template library). The iterator enables sequential and selective access of resources. For example, one can sequentially read the light intensity of each resource in the set R as follows:

```
Resource R;
Interator i;
Foreach i in R {
Print ("light intensity = %f\n", i ⟶ light);
}
```

However, the sequential readings cannot represent a snapshot of the desired target because the delay in accessing the whole set sequentially can be significant. In particular, the total delay is essentially the summation of all individual access time. Nevertheless, this individual approach is still useful, especially when only some elements in the set are accessed.

Parallel Access. Conversely, in this approach, all resources in the set are simultaneously accessed. This parallel access can be specified using a direct reference to the resource variable as follows:

```
Resouce R;
Printf ("light intensity =%f", R⟶ light);
```

In the above example, the program prints out the light intensity of all nodes in R. The total delay using this parallel approach is reduced to the longest delay of an access. The parallel approach not only reduces the total access time but also provides a much better snapshot of the desired target. Additionally, unlike the sequential approach, this parallel approach exposes an opportunity for the underlying system to perform in-network processing (e.g., data aggregation) that can significantly reduce a system's overall energy consumption [22–24, 27, 28]. An example of data aggregation

functions is max(A) whereby the maximum element in A is returned.

```
Resource R;
printf ("max light intensity = %f",
        max (R ⟶ light));
```

Ideally, the system expends energy only on delivering that max element, not on the others. This delivery can be practically approximated by in-network suppression of the elements whose values are less than that of the previously seen elements of the same access. Suppression will be ineffective or even impossible if the resources are accessed in sequence rather than in parallel.

4.4. Resource Binding. Our model supports two binding types: *dynamic* and *static*.

Dynamic Binding. In our paradigm, code does not need to be written to maintain binding between the physical resources and resource variables. Given that the resource property is constantly changing, rebinding the set of matching nodes is laborious. For example, the set of resources R at time t_1 can be completely different from the set of resources R at time t_2.

```
Resource R = ⟨expression1⟩
Time t1 = get_time ( );
x = Count (R);
. . .
Time t2 = get_time ( )
y = Count (R);
/∗Normally, x ! = y ∗/
```

Rather, it is desirable to simply provide the declarative expression that is associated with the resource variable to describe the resources of interest. In general, a reference to a resource variable implies a resource access. Our semantic of a resource-variable access is rather strict in a sense that the access is only performed on the resource that matches the declarative expression at the time of access. Furthermore, changes in the set of matching nodes do not require attention from programmers. (This is the main difference between our approach and the traditional approach that relies on node ids and OIDs of SNMP.) As a result, to conform with this strict semantic, the underlying system may need to spend significant overhead and excessive energy consumption for ensuring that this reactive binding is up to date. Therefore, we propose options or *tuning knobs* for lessening the semantic in order to save energy. For example, programmers can lessen the semantic by allowing access if the resource is bound in the last t seconds.

```
Resource R = ⟨expression,
                last_bound_time > now-t⟩
```

Furthermore, programmers can even specify an energy budget to bound the energy consumption of a resource access.

```
Resource R = ⟨expression,
                energy_budget = 100⟩
```

Other tuning knobs are currently under investigation.

Static Binding. Although the above dynamic binding of resources seems reasonable, one may notice that there are situations where dynamic bindings may not be appropriate. Specifically, we may want to access the previously matched resources that are no longer matched. For example, we may have turned on cameras in area A. However, after a period of time, we may want to turn them off, but some cameras have since been moved out of the area. If area A is included in our declarative expression, those cameras that have since been transferred will no longer match the expression. As a result, we may be unable to turn off the relocated cameras directly using the resource variable.

One solution to the above problem is to rely on the underlying system. For example, we could declare a new resource variable using a usual expression with an additional timing condition.

```
Resource R = ⟨expression1⟩;
Time t1 = get_time();
. . .
Resource X = ⟨expression1 && time == t1⟩;
```

As long as we know the time of the matching, we can describe the desired resources. A similar solution is to provide the function *last*() that returns the previous set of matching nodes to the caller. Therefore, we can operate on the desired set even though it no longer matches the expression.

```
Resource R = ⟨expression1⟩;
Resource X = last (R);
```

However, both solutions incur excessive overhead as the system is required to maintain all changes of a set at all times.

An alternative solution is to provide explicit instructions for memorizing matching nodes. We propose two explicit mechanisms: the *static* resource and the *iterator*.

Using the static resource variable, we can specify which resources are statically bound. The static resource variable will not be rebound in any circumstances. Therefore, we can maintain any set of resources even though they are no longer matched to the expression.

```
Resource R1;
Static Resource R2 = R1;
/∗ R1 changes over time but R2 does not∗/
```

This explicit instruction is cheaper to implement than last() because the system no longer has to keep all previous values of every resource variable. (Programmers might write "last(last(. . .last((R). . .)))".) Furthermore, the static resource is intended for memorizing the entire set of matching nodes. To memorize only one resource, an iterator is more appropriate. The value of an iterator does not automatically change without an explicit assignment.

```
Iterator i1 = R1 ⟶ first_element;
```

4.5. Access Timeout. Regardless of binding type, there is no guarantee that every WNES resource access will succeed. Unfortunately, WNES resource access time is unbound, and access failures are usually unavoidable because of network dynamics. Given that there is no response after unbound access time and failures, they cannot be easily differentiated. (This problem is similar to that of TCP. Packet loss and unbound acknowledgment delays are handled using time-out.) Timeout is usually a common technique for handling such problems. Therefore, we propose associating a resource variable with an access timeout. In this model, the access time is monitored for each access. Once an access has timed out, an exception is raised (similar to Java exceptions). It is necessary that the method for handling a time-out is explicitly specified in the *catch* statement.

```
Resource R = ⟨expression1, timeout = 10⟩
Iterator i = R ⟶ first_element;
try {
printf ("light intensity = %f", i⟶light);
} catch(TimeoutException) {
Printf ("cannot access the light sensor");
}
```

5. Implementation

There are currently two implementations (of all or part) of this abstraction: DRN on SM and *TinyDRN* on Maté. DRN on SM provides full features described in this paper. SM is a mobile-agent-based reprogramming middleware that runs on IPAQs equipped with 802.11 radios. We have also implemented *TinyDRN*, a bare subset of this abstraction on Maté. Maté is also a mobile-agent-based reprogramming middleware, but it runs on motes equipped with 802.15.4 radios. DRN and TinyDRN are implemented as libraries. Thus, there is no need for macrocompilers. In this section, we briefly overview the SM architecture and its advantages before we explain the detail of our implementation on SM and Maté.

5.1. Smart Message Architecture. Smart Messages (SMs) are mobile agents for wireless networks of embedded systems. They consist of code and data sections (called *bricks*), and a lightweight execution state. Unlike request/reply paradigms, SM applications need to migrate to nodes of interest and execute there. To do this, SMs execute a routing algorithm, carried as a code brick, for determining the next hop toward a node of interest. The code bricks are cached by nodes along the way to reduce the cost of transferring the same code in the future. Over time, this cost is amortized because of temporal and spatial locality of SM applications.

SM architecture consists of Smart Message Virtual Machine (SMVM) and Tag Space. SMVM is basically Sun's K Virtual Machine (KVM) that is modified to support Tag Space and program migration. SMVM is suitable for mobile devices with resource constraints and with as little as 160KB of memory [29]. Tag Space is a name-based memory region that unifies an interface to I/O and memory on SMVM. I/O and memory can only be accessed through an object called a *tag*. Direct-access instructions will not be recognized by SMVM.

Given that there is no Java thread or preemption in SM, the SM execution model is quite simple. Only one SM is active at a time on an SMVM while the other SMs wait in the queue until the active SM terminates, migrates, or suspends itself.

There are several advantages using SM as our target platform. One significant advantage is on-the-fly reprogrammability. New aggregation functions and predicates can be deployed on the fly. Another advantage is the SM tag, the unified interface to memory and I/O. This feature tremendously simplifies our implementation, especially our variable-like access to resource.

5.2. DRN Implementation Using SM. DRN is implemented as SM run-time libraries. Given that we only use regular SM commands to implement our libraries, we do not have to modify the SM Virtual Machine (SMVM) and do not have to implement a DRN macrocompiler. In the SM platform, each node is a pocket PC running an SMVM. Therefore, each node has interfaces to interact directly with a user. Consequently, we can use any node in the system as a user node.

Our macroprogram is implemented as a Smart Message that is injected at the user node. Generally, an SM program can migrate and execute at any node but, in our implementation, the main SM macroprogram do not migrate. To acquire data from other nodes, this main SM will create child SMs that migrate to those nodes and bring the data back.

When a resource variable is declared with a predicate and a binding lifetime, the main SM does not immediately bind the variable. Instead, the resource variable will be bound on demand when the variable is referred. Given that the set of matching nodes changes over time, binding the variable too early may not be useful. The variable is likely to be rebound at the time of access and overhead in early binding is wasted. This concept is similar to on-demand routing in wireless ad hoc networks.

To bind the variable, nodes that match the predicate have to be discovered (see Algorithm 1). The main SM creates a Discovery SM that contains the given predicate to discover nodes and their routes. The default target region is the whole network. If no target region is specified in the predicate, the Discovery SM floods the network by duplicating itself and migrating to all neighbors of the current node until all nodes are visited.

If there is geographical information about the target region in the predicate, the Discovery SM will migrate to only the neighbor that is closest to the target (Line 1–5). Upon reaching the region, the Discovery SM floods all nodes in the region to check if those nodes match the predicate (see Algorithm 2).

On each visited node, the Discovery SM creates a marking tag for differentiating visited nodes from others (Line 4). The Discovery SM will terminate if it arrives on a node with the marking tag (Line 1–3). In addition to the

```
1: while not in the target region do
2:    create marking tag
3:    migrate to the neighbor closest to the region
4:    create route-to-user tag point to previous node
5: end while
6: call Flood Migration
```

Algorithm 1: Resource discovery.

```
1: if marking tag exist then
2:    exit
3: end if
4: create marking tag
5: create route-to-user tag point to previous node
6: for each neighbor in the target region do
7:    create child Discovery SM with given predicate
8:    if this SM is the created child Discovery SM then
9:       migrate to the neighbor node
10:    call Flood Migration
11:    exit
12:   end if
13: end for
14: if this node matches the predicate then
15:    migrate back to user node
       (also create route-to-id and route
       to previous node along the way)
16:    add this node in a set of bound node
17: end if
```

Algorithm 2: Flood migration.

marking tag, the Discovery SM also creates a route-to-user tag on the current node for memorizing the previous hop (Line 5). Therefore, this route-to-user tag contains the next hop toward the user node. These route-to-user tags on all nodes form an aggregation tree for gathering data from all matching nodes.

Once a matching node is found, the Discovery SM migrates back along the aggregation tree to notify the main SM (Line 14–17). On the way back to the user node, the Discovery SM creates a route-to-id tag and a route-to-resource tag on the current node for memorizing the previous hop. Route-to-id tags form a path toward a matching node for sequential access whereas route-to-resource tags form a multicast tree for sending access request to matching nodes in parallel.

Upon reaching the user node, the Discovery SM notifies the main SM about the matching node. The main SM then adds the reported node into the set that is bound to the resource variable. Additionally, the main SM resets the binding timer of the resource variable to its lifetime.

In this implementation, an iterator access (i.e., sequential access) does not cause resource discovery. The main SM creates an Access SM that migrates toward the bound node using the corresponding route-to-id tags. Conversely, a resource-variable access causes resource discovery if the binding timer expires or the variable is not bound (see Algorithm 3). Once the variable is bound, the main SM creates Access SMs that migrate toward the bound nodes along the multicast tree in parallel (see Algorithm 4). Upon reaching the bound nodes, the Access SMs perform instructed operations and carry the results back to the user node (Line 1–8).

Accessing resources in parallel enables data aggregation that results in energy savings. Therefore, on each branching node along the aggregation tree, the Access SM waits for other SMs until SMs from all branches arrive or the waiting timer expires (Line 18). All arriving SMs are merged into one and only the resulting SM migrates to the user node (Line 19–23). Our waiting timer in this implementation is fixed and quite naive. A better implementation is to use the depth of the branching node to proportionally set its waiting timer. Undoubtedly, the deeper node requires the longer timer. The depth of the branching node can be computed when Discovery SMs migrate back to the user node. Given that a matching node is a leaf of our aggregation tree, its depth is zero. Each Discovery SM carries this depth counter and increments it by one for each hop that the SM migrates. The Discovery SM will also creates a depth tag on each node along the way for maintaining the current depth of the node if there is no such tag yet. Both the depth tag (on the current node)

```
1: if binding expired or not bound then
2:    call Resource Discovery
3:    restart binding timer
4: end if
5: call Access Migration
```

Algorithm 3: Resource access.

```
1:  if on a bound node then
2:    access the specified object
3:  end if
4:  if on a leaf node and exist parent node then
5:     migrate back to the parent node
6:     notify the parent SM
7:     return
8:  end if
9:  for each child on multicast tree do
10:    create child Access SM
11:    if this SM is the created child Access SM then
12:       remember this node as parent
13:       migrate to the child node
14:       call Access Migration
15:       exit
16:    end if
17: end for
18:  wait for results from all child Access SMs
19:  merge the results
20: if exist parent node then
21:     migrate back to the parent node
22:     notify the parent SM
23: end if
```

Algorithm 4: Access migration.

and the depth counter (on the SM) are compared and set to the greater value between them.

Furthermore, this implementation can also be improved by merging the Discovery SM and the Access SM into a Discovery&Access SM. This new SM behaves like the Discovery SM but, once it finds a matching node, it immediately accesses the resource on the node and migrates back to notify the main SM about the matching node as well as the access result in one step. This merging can improve energy savings and reduce delay of our system. We plan to implement this merging and depth computing in our future work.

5.3. TinyDRN. TinyDRN is a subset of our abstraction, retaining only resource variables and static binding features. TinyDRN is implemented as new bytecode instructions on Maté, a tiny virtual machine for sensor networks. In the Maté platform, each node is a mote running a Maté virtual machine. Given no user interface on motes, we need a PC as our user node that connects to a gateway mote for relaying our commands. Each sensor node or mote has upto 128 KB ROM for instruction memory and upto 4 KB RAM for data whereas the K Virtual Machine (used in SM) targets devices with a memory budget of at least 160 KB. As a virtual machine, Maté is a bytecode interpreter implemented as a component in Tiny OS (an operating system of motes).

In a sense, a Maté program is simply a script consisting of Maté commands that are recognized, interpreted, and executed by a Maté VM. To implement TinyDRN, we need to modify the Maté VM so that the virtual machine knows how to interpret and to execute our new bytecode instructions.

We have also developed an application using TinyDRN to test our TinyDRN implementation. Our testing application turns on the LED of nodes in the area that is brighter than 400 units. To understand the overhead of our implementation, we have written the same application using the original Maté (without TinyDRN instructions). Based on our measurement, this application with the original Maté takes 42,976 bytes of ROM and 3,134 bytes of RAM. In contrast, this application with the TinyDRN-added Maté takes 44,586 bytes of ROM and 3,289 bytes of RAM. The result indicates that the TinyDRN version takes only 3.75% additional bytes of ROM and 4.95% additional bytes of RAM.

Even with the slightly bigger memory usage, the TinyDRN version surprisingly runs faster and sends fewer messages than the non-TinyDRN one does. This is due to the smaller script size (a benefit of new bytecode instructions).

```
1:  Space sp=UNIVERSE;
2:  Resource R1=< (within(Sp)==TRUE)&(motion>0)  >;
3:  Location AverageLoc;
4:
5:  for(inti=1;i<= 25;i++) {
6:    AverageLoc = average(R1->Location);
7:    if (AverageLoc != NULL) {
8:      System.out.println(
9:          "Average("+i+")="+AverageLoc);
10:    sp.updateRegion(AverageLoc, 10);
11:   } else {
12:   System.out.println(
13:         "Average("+i+")= NOT FOUND");
14:   sp = UNIVERSE;
15:  }
16:  sleep(4000);
17:  }
```

Algorithm 5: Pseudocode for our object-tracking application.

Although the modified virtual machine is bigger, the application itself is smaller to propagate. This results in fewer messages and bytes to send over the network. Consequently, the code is propagated faster and the energy is consumed less. It is a classic example of using energy wisely on computation rather than on communication.

6. Evaluation

In this section, we conduct an experiment to evaluate a DRN application executed over our DRN runtime system. This section describes our methodology and considers the impact of a DRN tuning parameter on the application's performance.

However, only DRN is used in this evaluation because we intend to test the full feature of our abstraction. Our test on TinyDRN can be found in Section 5.3.

6.1. Goals, Metrics, and Methodology. We have implemented our object-tracking application (Section 6.2) using DRN. This application is evaluated on a network of 20 nodes. Each node is emulated using a Smart Message Virtual Machine (SMVM) that runs on a different port of a physical machine. (Given that the SMVM can run directly on an HP iPAQ [14], our DRN code can also run on the iPAQ without any modification.)

Our goals in conducting this evaluation study are twofold. First, it is necessary to verify the viability of the DRN model for macroprogramming WNES. Second, we would also like to understand the impact of resource-binding lifetime on the DRN application.

We choose two metrics to analyze the performance of our DRN application: *the number of application bytes sent* and *average distance error*. The number of application bytes that are sent measures the total bytes sent across the network. The metric roughly indicates the dissipated energy and implies the overall lifetime of WNES. Average distance error measures the distance between the actual object location and the reported location. This metric implies the accuracy of the tracking application; similar metrics were used in earlier work [30]. We study these metrics as a function of the resource binding's lifetime.

In our experiment, we study a multihop sensor field (of 20 nodes) that is generated by randomly placing the nodes in a 20 m by 40 m rectangle. Each node has a radio range of 10 m and a sensing range of 5 m. Such ranges enable a direct communication between two nodes that detect the same object. The transmission range also defines neighbors of each node (SMVM) in this emulation.

The DRN application tracks an object that moves at a rate of 0.25 m/s. The object moves clockwise along the edge of a 10 m by 30 m rectangle located in the middle of the sensor fields. This clockwise movement causes nodes in different regions to detect and track the object. The application estimated the object location on 25 different occasions during our experiment, or once every 4 seconds.

6.2. Object Tracking Application. Algorithm 5 shows the simple DRN pseudocode that accompanies our object tracking application. Essentially, the application tracks an object by acquiring the location of devices (i.e., resources) that detect motion within a region of interest. The average location of such devices is an estimation of the object location. At the beginning, there is no estimation of object location. The application first searches for the object throughout the sensor field. Once an object location is found, the region of interest for the next search is set to an area within 10 m of the estimated location. This approach limits the searching space and results in better energy efficiency, especially when the geographical routing is used in the underlying system. Later, if the object cannot be found in this dynamic circular region, the region of interest is reset to the whole sensor field.

The actual Java code for this application (listing 1) is very similar to the simple DRN pseudocode in Algorithm 5;

```
1: public class TrackingApp extends SmWrapper{
2:
3:   private final static int timeout = 24000; // Binding lifetime
4:   private Space sp;
5:   private TrackingExpression tExp;
6:   private Resource resource;
7:   private LocationAverage agg;
8:
9:   public TrackingApp(){
10:    super("TrackingApp");
11:  }
12:
13:  public void run() {
14:    try{
15:      sp = new Space(null, -1); // sp = UNIVERSE
16:      tExp = new TrackingExpression(sp, ''motion'');
17:      resource = new Resource(tExp, timeout);
18:      agg = new LocationAverage();
19:       for(inti=1;i<= 25; i++){
20:         agg = (LocationAverage)resource.access(agg, 4000);
21:         System.out.println("agg = "+agg);
22:          Location average = (Location)agg.evaluator();
23:          if (average != null){
24:           System.out.println("Average("+i+") = "+average);
25:           sp.updateRegion(average, 10);
26:         }else{
27:           System.out.println("Average("+i+") = NOTFOUND");
28:           sp.updateRegion(null, -1); // sp = UNIVERSE
29:         }
30:          sleep(4000);
31:       }
32:    } catch(Exceptione){}
33: }
34:
35: public static void main(String[]args){
36:    TrackingApp trackingApp = new TrackingApp();
37:    String[]types;
38:    types = new String[3];
39:    types[0] = "TrackingApp";
40:    types[1] = "TrackingExpression";
41:    types[2] = "LocationAverage";
42:    trackingApp.initSM(types,trackingApp);
43:    trackingApp.run();
44: }
45: }
```

Listing 1: Real Java code for our object-tracking application.

it is possible to achieve a one-to-one translation from simple DRN pseudocode to real Java code. In this Java code, our *TrackingApp* simply extends the *SmWrapper* that hides SM-related details from programmers. To conform with the Java syntax, we implement the resource expression (*TrackingExpression*) as a class (listing 2). (Automatic generation of this expression class from DRN pseudocode is part of our future work.) Each expression class contains an *evaluate()* method that needs to be executed on the device to determine if the device property is matched with the expression.

In this application code, resources are accessed in parallel. Parallel access provides an opportunity for in-network processing (e.g., data aggregation) that can significantly reduce the system's overall energy consumption [22–24, 27, 28]. Typically, in other systems, the code for in-network aggregation cannot be dynamically installed after network deployment. In some systems, an API may not be provided for writing a new in-network aggregation code. For example, it is not obvious how a new aggregation algorithm can be expressed in TAG using SQL, given that SQL is not designed for expressing algorithmic details. Furthermore, TAG is not

```
1:  public class TrackingExpression extends Expression {
2:
3:    private Space sp_;
4:    private String moTag_;
5:
6:    public TrackingExpression (Space sp, String moTag) throws BadSMApiUsageException  {
7:        sp_=sp;
8:        moTag_=moTag;
9:    }
10:
11:   public boolean evaluate()  {
12:       try {
13:           Integer moInt = (Integer)TagSpace.readTag(moTag_);
14:           GPSData gps = (GPSData)TagSpace.readTag("gps");
15:           if (!sp_.outside(new Location(gps.latitude, gps.longitude)) && (moInt.intValue() >0) )  {
16:               return true;
17:            }
18:       } catch(Exception e) {}
19:       return false;
20:   }
21: }
```

Listing 2: TrackingExpression class for matching resources.

a reprogrammable platform. Therefore, it is not clear how a new aggregation code can be deployed on the fly. Unlike other WNES programming approaches, DRN provides an *Aggregation* class that can be extended to implement a new dynamically-deployable aggregation technique.

Generally, a data aggregation technique is implemented using three functions: initializer *i()*, merger *m()*, and evaluator *e()*. The initializer *i()* specifies how to instantiate a data state record for a single sensor value. DRN will call this function on devices whose properties are matched with the declarative expression. This data state record will then be sent back toward the user node. During the return trip, this data state may meet other data states from the same set of desired resources. DRN will call the merger *m()* to aggregate these data states into one. Once the data state reaches the user node, the evaluator *e()* will compute the actual value of the aggregate.

In our application, we have shown how to implement a new aggregation technique called *LocationAverage* (listing 3) in DRN. To do this, we simply extend the *Aggregation* class and overload the three mentioned functions. We use $\langle \text{sum\_}x, \text{count\_}x, \text{sum\_}y, \text{count\_}y \rangle$ as our data state. Suppose the matching device is located at $(x1, y1)$. The initializer sets the data state record to $\langle x1, 1, y1, 1 \rangle$. The merger combines the state $\langle x1, cx1, y1, cy1 \rangle$ and the state $\langle x2, cx2, y2, cy2 \rangle$ into a single state $\langle x1 + x2, cx1 + cx2, y1 + y2, cy1 + cy2 \rangle$. The evaluator returns $\langle \text{sum\_}x/\text{count\_}x, \text{sum\_}y/\text{count\_}y \rangle$ as the average location.

6.3. Tuning Knob. Semantically, in our model, resource access is strictly performed on resources that match the declarative expression at the time of access. Changes in the set of matching nodes do not require attention from the programmers. Therefore, DRN must rebind resources transparently and dynamically. This strict semantic could incur significant overhead and excessive energy consumption for ensuring that this reactive binding is up to date. Not surprisingly, we propose tuning knobs for balancing strong semantics with energy savings. One of these tuning knobs is the *resource binding lifetime*. For example, using a binding lifetime of t, programmers can slightly lessen the semantic and allow access if the resource is bound in the last t seconds.

In this experiment, we study an impact of binding lifetime on energy consumption and tracking accuracy of an unoptimized version of our application. Specifically, Line 25 in listing 1 is removed. Therefore, searches for the object are always performed throughout the sensor field. An additional objective of this experiment is to show that, even though the declarative expression and related variables are not changed, the resource is dynamically and deservedly rebound.

Figure 1(a) plots the number of bytes sent as a function of the resource binding lifetime. As expected, the number of bytes sent is reduced (the line with black rectangles) as we increase the binding lifetime (i.e., reduce the number of resource discovery). Results indicate that it is possible to achieve meaningful energy savings without a significant degradation in tracking accuracy. Specifically, we can achieve a 51.5% savings in bytes sent with only small accuracy degration when we increase the binding lifetime from 4 to 16 seconds. The total number of bytes sent includes the overhead for installing this mobile-agent program on the fly (the line with white rectangles). When we factor out the bytes sent for injecting the application code into the network, the savings improve to 55.2%.

The average tracking error does not significantly increase until the binding lifetime is more than 16 seconds

```
1:  public class LocationAverage extends Aggregate {
2:
3:    private GPSData gps;
4:    double sum_x, sum_y;
5:    int count_x, count_y;
6:
7:   public void initializer()  {
8:       try {
9:           gps = (GPSData)TagSpace.readTag("gps");
10:           sum_x = gps.latitude;
11:          sum_y = gps.longitude;
12:          count_x = 1;
13:          count_y = 1;
14:      }  catch (Exception e) {}
15:   }
16:
17:   public void merger(Aggregate agg)  {
18:       sum_x = sum_x+agg.sum_x;
19:       sum_y = sum_y+agg.sum_y;
20:       count_x = count_x+agg.count_x;
21:       count_y = count_y+agg.count_y;
22:   }
23:
24:   public Object evaluator()  {
25:        try {
26:           return new Location(sum_x/count_x, sum_y/count_y);
27:        }catch (Exception e)  {
28:           return null;
29:        }
30:   }
31:  }
```

Listing 3: LocationAverage class for in-network processing.

(Figure 1(b)). The result is intuitive. If the object moves away from a bound sensor at the speed of 0.25 m/s, it will take at most 20 seconds to move beyond the bound node's sensing range. Conversely, if the object moves toward the bound sensor without changing its direction, it will take at most 40 seconds to pass out of range. Given the moving pattern in this experiment, we do not need to rediscover the resources within 20 seconds to achieve a reasonable accuracy. However, after 20 seconds, the accuracy will be significantly degraded. If we do not rediscover the resources after 40 seconds, we will no longer be able to track the object.

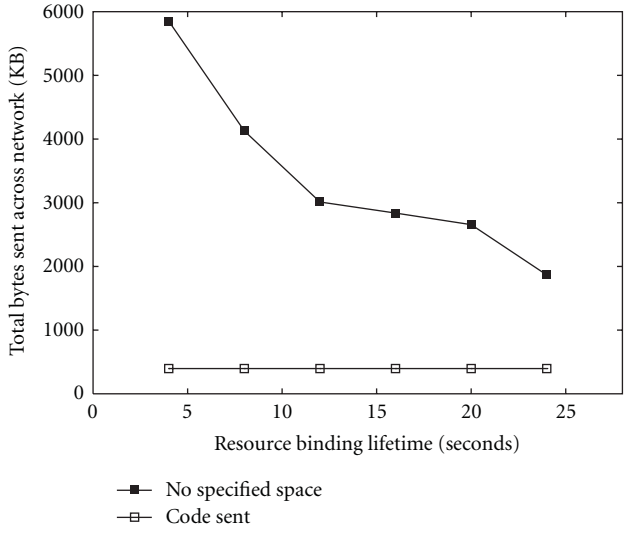
Tracking accuracy depends on several factors: estimation techniques, network density, and sensing range. The estimation error of 2-3 m in this experiment is considered reasonable, given our simple estimation technique, low-density network, and 5 m sensing range.

6.4. Space Scoping for Optimization. Like other programming paradigms, writing an efficient program requires understanding of the underlying system. For example, in virtual memory systems, programs should be written such that the number of page faults is minimized. To operate on an entire two-dimensional array in those memory systems, elements in the array should be accessed row-by-row rather than column-by-column. Similarly, our tracking application is more efficient when the searching space is specified because our run-time library supports geographic routing. Given a specified space, resource-discovery request is geographically routed to the space instead of flooding throughout the network.

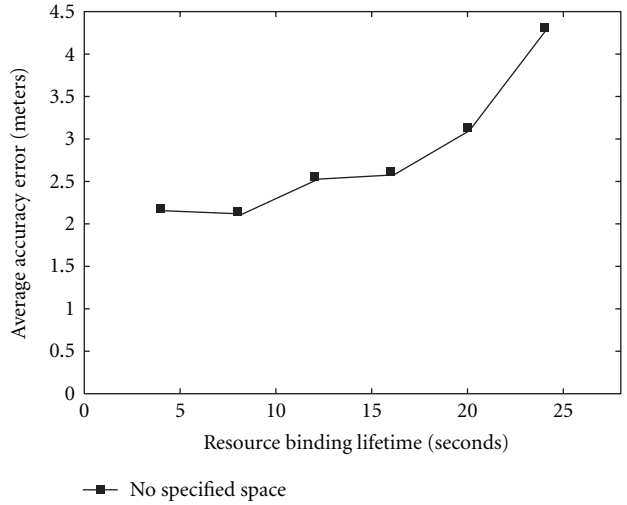
To study the impact of space scoping on our tracking application, we conduct an experiment similar to that of the previous section. The difference is that Line 25 in listing 1 is now included.

As the binding lifetime is increased, the savings is decreased due to the reduced number of resource discovery. Additionally, the tracking accuracy is not significantly degraded by space scoping (Figure 2(b)).

Our results indicate that we can achieve 42.5% savings on the number of bytes sent when we dynamically specify the target space (Figure 2(a)). Although this savings is significant, one may expect more savings because geographic routing is much more efficient than flooding. However, once the resource-discovery request is geographically routed to the specified space, the request is flooded within the space in order to discover all matching nodes. This scoped flooding incurs additional overhead and results in fewer-than-expected savings.

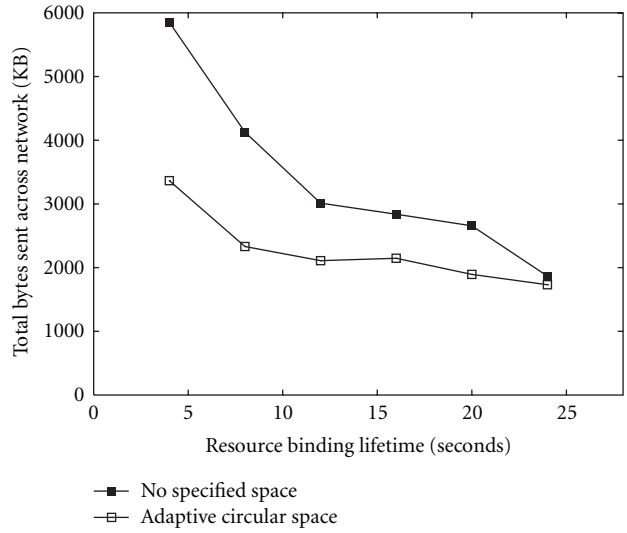


(a) Application bytes sent.

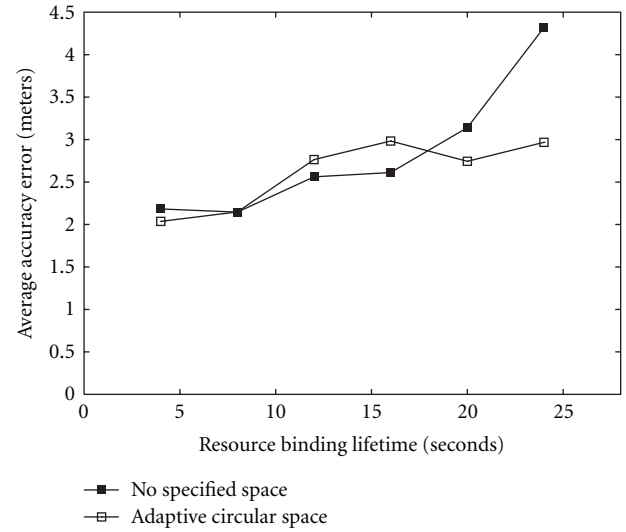


(b) Average distance error.

Figure 1: Impact of resource binding lifetime on our object-tracking application.



(a) Application bytes sent.



(b) Average distance error.

Figure 2: Impact of space scoping on our object-tracking application.

7. Discussion

In Section 6, our tracking example is intentionally simple. Undoubtedly, one can easily write a more sophisticated tracking application using DRN. One possible improvement is to estimate the object speed using exponential weighted average. This estimated speed can be used to predict the object location or the center of the space for the next access.

To simplify our evaluation, we have used fixed sensing and radio ranges. In practices, these ranges are not fixed. With realistic ranges, we can obtain more realistic accuracy results. However, these ranges have no impact on the viability of our abstraction.

While DRN works well for tracking, some may wonder if it applies well to other applications, especially data-collection applications. It is necessary to emphasize that DRN is not only useful for retrieving data but also for pushing data to declaratively named nodes. We can install a new function to all nodes in the network. This new function then reads and pushes data periodically or reactively toward the user node. Therefore, data-collection applications should be easily implemented by using DRN in this manner.

In this paper, we propose only two tuning knobs: binding lifetime and access timeout. There are still several other tuning knobs that should be exposed to programmers. For example, one might want to dynamically change the number of nodes that participate in aggregation based on energy goals. Such a tuning knob would be useful for limiting energy consumption during resource access. We plan to explore other tuning knobs in our future work.

Furthermore, one may wonder what the right binding lifetime is for each application. As long as the result is still acceptable, higher binding lifetime is generally better.

A possible solution without trial-and-error is to dynamically adjust the binding lifetime based on the quality of the result and the resource consumption.

8. Conclusions

We believe that, to efficiently develop WNES applications, appropriate programming abstractions are necessary. DRN is one such abstraction that integrates declarative constraints with imperative constructs to form a powerful programming paradigm suitable for macroprogramming WNES.

We have implemented DRN on two platforms: SM and Maté. SM can run on iPAQs with 802.11 radios whereas Maté can run on motes with 802.15.4 radios. Furthermore, given network transparency, our approach should be applicable for macroprogramming over wired or wireless networks as well. However, this network transparency feature of DRN implies that DRN is not for low-level programming or implementing a protocol that requires a distinct notion between being remote and local.

In addition, we have implemented an object-tracking application using our DRN runtime library to show the model viability. We have also evaluated our DRN runtime library and its tuning knob (i.e., resource binding lifetime). Our tuning knob enables the DRN application to save up to 55.2% of bytes sent without significant accuracy degradation when the application code is already cached or installed in the network.

In the future, we intend to further explore the design space of DRN such as other tuning knobs. Additionally, we plan to implement other applications using DRN and to conduct more extensive evaluation in order to better realize DRN's full potential.

Acknowledgments

This research has been supported by the Thailand Research Fund (TRF) under Grant no.MRG5080449. The authors would like to thank Napat Ratanasirintrawoot and Phuri Chalermkiatsakul on their work on TinyDRN and also thank Prof. Rajesh Gupta, Amin Vahdat, and anonymous reviewers for valuable suggestions to improve this work.

References

[1] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: an acquisitional query processing system for sensor networks," *ACM Transactions on Database Systems*, vol. 30, no. 1, pp. 122–173, 2005.

[2] P. Bonnet, J. Gehrke, T. Mayr, and P. Seshadri, "Query processing in a device database system," Tech. Rep. TR99-1775, Cornell University, 1999.

[3] Y. Yao and J. Gehrke, "The cougar approach to in-network query processing in sensor networks," *SIGMOD Record*, vol. 31, no. 3, pp. 9–18, 2002.

[4] S. Choochaisri, N. Pornprasitsakul, and C. Intanagonwiwat, "Logic macroprogramming for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2012, Article ID 171738, 12 pages, 2012.

[5] S. Choochaisri and C. Intanagonwiwat, "A system for using wireless sensor networks as globally deductive databases," in *Proceedings of the 4th IEEE International Conference on Wireless and Mobile Computing, Networking and Communication (WiMob '08)*, pp. 649–654, IEEE Computer Society, Washington, DC, USA, October 2008.

[6] R. Gummadi, O. Gnawali, and R. Govindan, "Macroprogramming wireless sensor networks using Kairos," in *Proceedings of the 1st IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '05)*, pp. 126–140, July 2005.

[7] The Castle project, http://www.cs.berkeley.edu/projects/parallel/castle/split-c/.

[8] C. Borcea, C. Intanagonwiwat, P. Kang, U. Kremer, and L. Iftode, "Spatial programming using smart messages: design and implementation," in *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS '04)*, pp. 690–699, Tokyo, Japan, March 2004.

[9] L. Iftode, C. Borcea, A. Kochut, C. Intanagonwiwat, and U. Kremer, "Programming computers embedded in the physical world," in *Proceedings of the 9th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS '03)*, San Juan, Puerto Rico, May 2003.

[10] R. Newton, G. Morrisett, and M. Welsh, "The regiment macroprogramming system," in *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks (IPSN '07)*, pp. 489–498, ACM, New York, NY, USA, April 2007.

[11] T. W. Hnat, T. I. Sookoor, P. Hooimeijer, W. Weimer, and K. Whitehouse, "Macrolab: a vector-based macroprogramming framework for cyber-physical systems," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys '08)*, pp. 225–238, New York, NY, USA, 2008.

[12] Y. H. Tu, Y. C. Li, T. C. Chien, and P. H. Chou, "EcoCast: interactive, object-oriented macroprogramming for networks of ultra-compact wireless sensor nodes," in *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '11)*, pp. 366–377, Chicago, Ill, USA, April 2011.

[13] C. Intanagonwiwat, R. K. Gupta, and A. Vahdat, "Declarative resource naming for macroprogramming wireless networks of embedded systems," in *ALGOSENSORS*, S. E. Nikoletseas and J. D. P. Rolim, Eds., vol. 4240 of *Lecture Notes in Computer Science*, pp. 192–199, Springer, 2006.

[14] C. Borcea, D. Iyer, P. Kang, A. Saxena, and L. Iftode, "Cooperative computing for distributed embedded systems," in *Proceedings of the 22nd International Conference on Distributed Systems (ICDCS '02)*, pp. 227–236, July 2002.

[15] A. Boulis, C. Han, and M. Srivastava, "Design and implementation of a framework for efficient and programmable sensor networks," in *Proceedings of the 1st International Conference on Mobile Systems, Applications, and Services (Mobisys '03)*, pp. 187–200, San Francisco, Calif, USA, May 2003.

[16] P. Levis and D. Culler, "Maté: a tiny virtual machine for sensor networks," in *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (APLOS '02)*, pp. 85–95, October 2002.

[17] K. Whitehouse, F. Zhao, and J. Liu, "Semantic Streams: a framework for composable semantic interpretation of sensor data," *Wireless Sensor Networks*, vol. 3868, pp. 5–20, 2006.

[18] S. Choochaisri and C. Intanagonwiwat, "An Analysis of Deductive-Query Processing Approaches for Logic Macroprograms in Wireless Sensor Networks," *Engineering Journal*, vol. 16, no. 4, pp. 47–62, 2012.

[19] Oracle Corporation, *Pro∗c/c++ Precompiler Programmer's Guide Release 9.2*, 2002.

[20] M. Grabmuller, *Constraint imperative programming [Diploma thesis]*, Technische Universitat Berlin, 2003.

[21] S. Chen, P. B. Gibbons, and S. Nath, "Database-centric programming for wide-area sensor systems," in *Proceedings of the 1st IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '05)*, pp. 89–108, July 2005.

[22] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, "Building efficient wireless sensor networks with low-level naming," in *Proceedings of the ACM Symposium on Operating Systems Principles*, Banff, Canada, October 2001.

[23] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM '00)*, pp. 56–67, Boston, Mass, USA, August 2000.

[24] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 2–16, 2003.

[25] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Siences*, p. 223, Maui, Hawaii, January 2000.

[26] D. Coffin, D. Van Hook, R. Govindan, J. Heidemann, and F. Silva, "Network routing application programmer's interface (api) and walk through 8.0," Tech. Rep. 01-741, USC/ISI, 2001.

[27] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," in *Proceedings of the 22nd International Conference on Distributed Systems*, pp. 457–458, IEEE, Vienna, Austria, July 2002.

[28] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks," in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI '02)*, December 2002.

[29] C. Borcea, C. Intanagonwiwat, A. Saxena, and L. Iftode, "Self-routing in pervasive computing environments using smart messages," in *Proceedings of the 1st IEEE International Conference on Pervasive Computing and Communications (PerCom '03)*, pp. 87–96, March 2003.

[30] M. Welsh and G. Mainland, "Programming sensor networks using abstract regions," in *Proceedings of the 1st USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI '04)*, March 2004.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 785984, 19 pages
doi:10.1155/2012/785984

Research Article

Human-Mobility-Based Sensor Context-Aware Routing Protocol for Delay-Tolerant Data Gathering in Multi-Sink Cell-Phone-Based Sensor Networks

M. B. Shah, S. N. Merchant, and U. B. Desai

Department of Electrical Engineering, Indian Institute of Technology Bombay, Powai, Mumbai 400076, India

Correspondence should be addressed to M. B. Shah, mbshah@iitb.ac.in

Received 17 February 2012; Revised 20 June 2012; Accepted 25 July 2012

Academic Editor: Sherali Zeadally

Ubiquitous use of cell phones encourages development of novel applications with sensors embedded in cell phones. The collection of information generated by these devices is a challenging task considering volatile topologies and energy-based scarce resources. Further, the data delivery to the sink is delay tolerant. Mobility of cell phones is opportunistically exploited for forwarding sensor generated data towards the sink. Human mobility model shows truncated power law distribution of flight length, pause time, and intercontact time. The power law behavior of inter-contact time often discourages routing of data using naive forwarding schemes. This work exploits the flight length and the pause time distributions of human mobility to design a better and efficient routing strategy. We propose a Human-Mobility-based Sensor Context-Aware Routing protocol (HMSCAR), which exploits human mobility patterns to smartly forward data towards the sink basically comprised of wi-fi hot spots or cellular base stations. The simulation results show that HMSCAR significantly outperforms the SCAR, SFR, and GRAD-MOB on the aspects of delivery ratio and time delay. A multi-sink scenario and single-copy replication scheme is assumed.

1. Introduction

Cyber Physical Systems (CPSs) [1, 2] are prospective technologies of the next millennium, which encompasses the vision of merging communication, computing and control systems in such a manner that they become almost invisible to layman's life and still enhance their living quality. CPS, are a paradigm shift in the conventional object-oriented thought process. CPS enables many machine-machine, machine-human, and human-human interactions, both in physical and cyber world. CPS-based services such as traffic control, pollution control, automated highways, automatic chemical processes, and power grid control need extensive monitoring in real world. In order to capture physical reality into decision making and controlling aspect, CPS needs to interact with Wireless Sensor Networks (WSNs). However, lack of ubiquitousness of WSN prevents penetration of CPS systems. In such a scenario, cell phones that are now equipped with sensory functionalities such as Global Positioning System (GPS), audio processing, camera, pollution sensors, and accelerometers can be exploited. With the recent transition from the feature to the smart phone, the mobile phone has now extensive sensing capabilities and can work as a mobile sensing device [3]. Moreover, the number of smart phones has been rapidly increasing making cell phones an ideal ubiquitous platform.

Mobile sensing with smart phones has been explored extensively [4–9]. The pollution monitoring application using Cell-Phone-based Sensor Network (CPSN) developed in [10] uses short-range communication outlets such as Wi-Fi or Bluetooth. In this paper, we consider the same CPSN architecture as of [10] shown in Figure 1.

CPSN has many challenges such as volatile topology, limited buffer space and loose connectivity with neighboring nodes. Data gathering protocols used for traditional WSN are ineffective due to aforementioned issues. Albeit the cell phone user's mobility, enables a CPSN node to come in contact with other node and create opportunities to encounter any useful forwarder which can get their messages closer to its intended destination even in the absence of

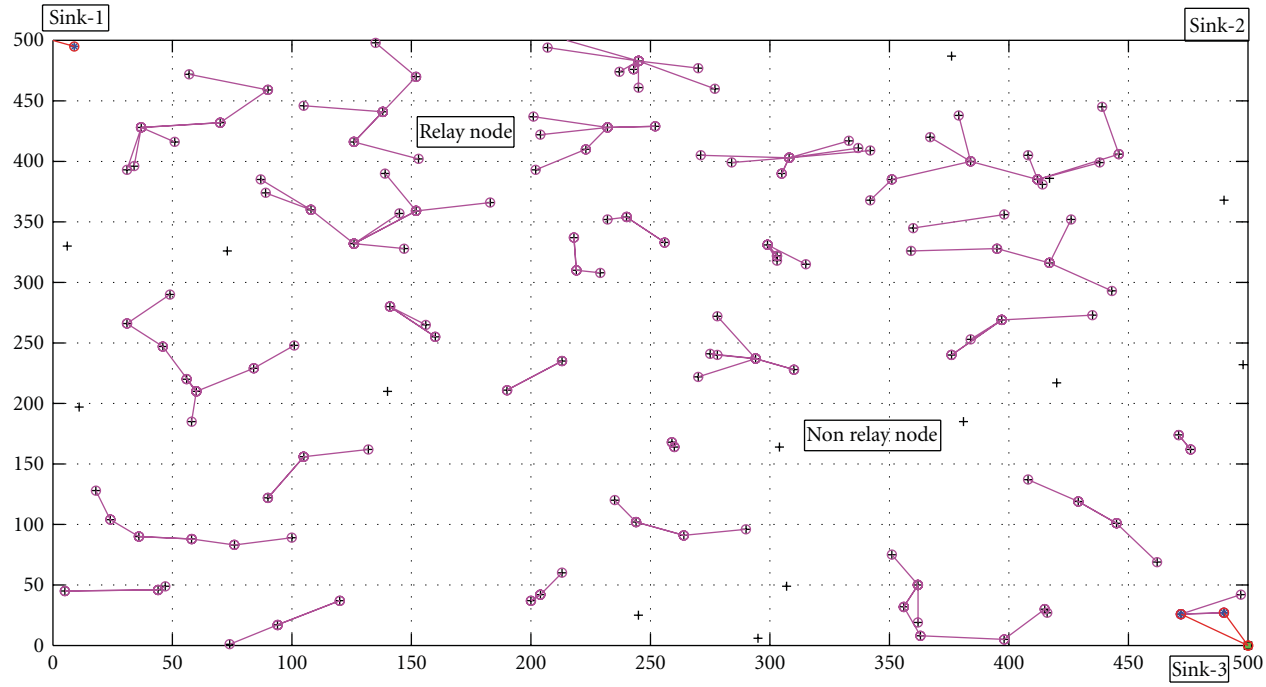


Figure 1: Cell-phone-based sensor network.

an end-to-end path [11]. CPSN can exploit these human interactions opportunistically to route the data to the sink or Base Station (BS). We assume CPSN as a type of delay-tolerant, intermittently connected, or opportunistic network [12–14] where end-end path is not available.

The common approach for data gathering in opportunistic sensor networks is directed transmission, where data is delivered to the sink when it is in its direct proximity. To be effective this scheme needs multiple sinks [15]. Another is flooding where messages are forwarded in multiple copies hoping to reach at the sink as early as possible. The third scheme is delay and fault-tolerant data delivery scheme (DFT-MSN) [16], where selected nodes are used as a forwarder of data messages. For CPSN environment, it is preferable to use this approach as energy consumption needs to be minimized.

The opportunistic data forwarding approaches [17, 18] identifies certain lead nodes that act as the relay point of communication between nodes and to be effective, they need good connectivity. These routing protocols select their relay nodes in such a way that the data packets are delivered using minimum hops to the destination. Connection and disconnection with destination node or sink, due to mobile nature of cell phone users, decide the protocol's performance. The mobility of cell phone users cannot be controlled, so the need is to select or predict an appropriate relay node, otherwise the overhead of forwarding data to wrong relay nodes will result in high energy expenditure and negate the benefits of opportunism.

Routing is an NP-complete problem, and a significant amount of research focuses on optimal methods for routing nodes in delay-tolerant Networks (DTNs) [19–24]. This DTN network scenario is of unicast transmission where message data are sent between pair of source and destination node, while the CPSN scenario is of convergecast transmission where all message data has common destination. Compared with DTN, new opportunities exist in CPSN for relay selection due to increased node density. Furthermore, the human mobility patterns show specific human walk characteristics; namely, truncated power law distributions of flight length and pause time [25]. Given this mobility pattern, it is possible to relay or route more intelligently and design a better routing method for cell phone environments.

Several methods were proposed for forwarding data in mobile sensor networks. Routing approach named Sensor Context Aware Routing (SCAR) [26, 27] uses prediction technique over the context of a mobile sensor node (such as previously encountered history with sink nodes, energy level of each node, and connectivity difference with neighboring nodes) to forecast the best relay node from the neighboring nodes. It uses Kalman-filter-based prediction approach. SCAR uses a replication scheme by which the master copy of the message is retained in the network till it reaches the sink. Secondary copies, sent to the relays, are deleted if buffer becomes full.

Recently authors of [28] have proposed the scheme named gradient-based routing approach incorporating node Mobility (GRAD-MOB). Performance is improved by predicting relay node based on their relative movement with respect to sink. The node moving away from sink is not preferred as a relay node.

The approach of Scale Free Routing (SFR) [29] improves performance with the use of ballistic nodes as relay nodes. The ballistic nodes exhibit sudden jumps of flight length

as per TLW mobility model. Use of these nodes as relays, allows message data to be forwarded from the low to high utility area near sink. Our approach is to prevent nodes from forwarding messages in low utility area with a newly defined utility function.

Motivated by the SCAR and the SFR, we propose a Human Mobility Sensor Context-Aware Routing (HMSCAR) protocol tailored for CPSN architecture with multiple sinks. HMSCAR incorporates two new metrics based on human walk characteristics to predict best relays. The new parameters introduced in our algorithm is based on human walk characteristics; namely, truncated power law distributions of the flight length and the pause time. Use of these parameters is justified for cell phone-based wireless sensor networks as cell phone users exhibit human walk characteristics. SCAR assigns weights to mobile nodes based on metric-like change of connectivity, battery level, and collocation with sink. The newly introduced parameters operate on decision of super flight length and super pause time. A given flight length is super flight or super pause if the flight length or pause time is above certain threshold. The proposed algorithm is compared with SCAR, GRAD-MOB, and SFR using greedy prediction approach where node with highest weight value is preferred as the next relay. Moreover, we evaluate the performance of our algorithm with single copy replication scheme assuming single sink and multisink presence. Single copy replication schemes are basis of multicopy replication schemes. Relay node discard packets if its buffer becomes full.

In each sink or base station initiated round of data collection, the selected relays receive data samples from their surrounding members by 802.11-based Request to Send and Clear to Send (RTS-CTS) mechanism and store them in buffers. The stored data is again forwarded to the selected relays in next round of operation. The relay node uses a bloom filter to compress the data into L bits before it forwards to another relay in the path. Nodes that are in direct connection with the sink will finally send their stored data to it. Use of routing table at each node enables forwarding of message data from lowest utility node to highest utility nodes in each round of data collection.

We have presented the related work in Section 2. In Section 3, we discuss the system model. In Section 4, we introduce a definition of Super Pause Time and Super flight length and propose the HMSCAR algorithm. The performance metrics and simulation setup is described in Sections 5 and 6, followed by the results and conclusion in Sections 7 and 8, respectively.

2. Related Work

There are many approaches proposed for routing in Delay Tolerant Networks (DTN). A detailed survey of all these techniques has been done in [23, 24]. Forwarding strategies of DTN are either epidemic [21, 22] or probabilistic [19]. Epidemic protocols use varieties of replication schemes for reducing packet delay in the network at the cost of energy consumption. The probabilistic methods make smart decisions of packet forwarding so that the packet delivery ratio is increased. It is very difficult to achieve both high delivery ratio and low delivery delay for a given energy and storage constraints [23]. Probabilistic methods leverage on a utility function, which is either destination dependent, destination independent, or hybrid. The SCAR algorithm proposed in [26] is a probabilistic one and based on hybrid utility function. In [30] a content- and context-aware routing protocol (CCBR) for multi-sink scenario is proposed. CCBR provides good delivery ratio with low power consumption. In [31] a history-based routing approach named CHARON is presented, which estimates delivery delay (EDD) at each node and routes the messages towards the sink along decreasing delay gradient. CHARON utilizes EDD as a context parameter, instead of traditional metrics such as relative mobility or sink encounter frequency. It performs better with low-density network scenario. In [32] a DTN protocol called SMITE is proposed for efficient data delivery of data packets to a mobile sink. SMITE outperforms SCAR, DFT-MSN and sidewinder [33] on the aspect of delivery ratio, transmission overhead, and time delay. SMITE uses cluster-based aggregation mechanism coupled with angle-based transmission mechanism to deliver data to the mobile sink.

The authors of [34] propose a mechanism to calculate delivery probability based on node's moving direction, speed, and distance between node and sink. Similar approach called as GRAD-MOB is applied in [28], where in addition to gradient-based routing metric another metric, based on node's moving direction from the sink node, has been added. This algorithm has novelty of incorporating node mobility for DTN networks. All above approaches use either single copy, multi-copy, or hybrid replication scheme. Contrary to this, the authors in [35] has proposed a DTN routing protocol, which uses repetitive contacts and their time sequence. The protocol does not replicate message and hence can work for lower buffer size of mobile nodes.

Human-carried mobile devices have some specific movement patterns in real life. Some of the movement patterns have been studied in [36, 37]. In [11], the effect of human mobility was evaluated on two-hop relaying algorithm, which was proposed in [38]. The authors of [11] have analyzed the performance of two-hop relaying protocol in a network of nodes following power law intercontact time. Result indicates infinite delay if power exponent is greater than two, which is the case of light tail. The delay becomes finite if multicopy replication approach is used.

Authors of [25] have proposed Truncated Levy Walk (TLW) mobility model. The mobility traces generated by TLW with various combinations of levy exponent α and β fit real human walk traces for various outdoor settings. In [29] a Scale-Free Routing (SFR) scheme is proposed based on observation of flight length characteristics of TLW. Some special nodes called ballistic nodes or super nodes are identified and messages are forwarded, if such nodes are available as relays. The SFR claims to achieve lower delivery delay compared to traditional gradient-based utility schemes, which may lead to local minima. SFR approach is generalized and can be applied with any gradient-based scheme. However, authors of [29] have made assumptions

of knowing mobility information of ballistic node, which is a strong one, and have proposed secondary means to predict it. Our work is also based on observation of TLW mobility model. We leverage both pause time and flight length distributions of human walk without using secondary means to predict it. Our proposed scheme HMSCAR is based on properties of heavy-tailed data called declining hazard rate and mass-count disparity [39]. We use a threshold-based approach to predict about super flight length or super pause time. Prediction of super flight length is based on declining hazard rate property of heavy-tailed data and that of super pause time is based on mass-count disparity property of heavy-tailed data.

The use of mobile phone for sensing was advocated in [40, 41]. Authors of [42] have used a layered approach for data monitoring application in multihop cell-phone based Sensor Network (MSCN). We propose a cluster-based data-gathering protocol for MCSN and Single-hop Cell-phone-based Sensor Network (SCSN) [43, 44], where the data gathering schemes are designed for nondelay tolerant monitoring application.

3. System Model

Mobility models emulate the behavior of real life mobile user as closely as possible. Truncated Levy Walk (TLW) mobility model proposed in [25] is based on, about one thousand hours of GPS traces, involving 44 volunteers in various outdoor settings. The TLW model captures movement of the people in various outdoor scenarios such as campus area, theme park, fair, and metropolitan area. As the cell phone users exhibit human walk characteristics, we have used TLW model as the mobility model for performance evaluation of HMSCAR.

3.1. Truncated Levy Walk Mobility Model. Recent findings of human walk characteristics show truncated power law distribution of flights (straight line trip without directional change), pause times, and intercontact times (the time elapsed between two successive contacts of the same pair of nodes). Figure 2 shows mobility traces of TLW mobility model.

(i) Flight lengths follow a truncated power law

$$P(l) \propto |l|^{-(1+\alpha)}, \quad 0 < l < l_{\max}, \tag{1}$$

where $l_{\max}$ is the maximum flight length.

(ii) Pause times follow a truncated power law

$$\phi(t) \propto t^{-(1+\beta)}, \quad 0 < t < t_{\max}, \tag{2}$$

where $t_{\max}$ is the maximum pause time.

(iii) Turning angles follow a uniform distribution.

(iv) Velocity increases as flight lengths increase.

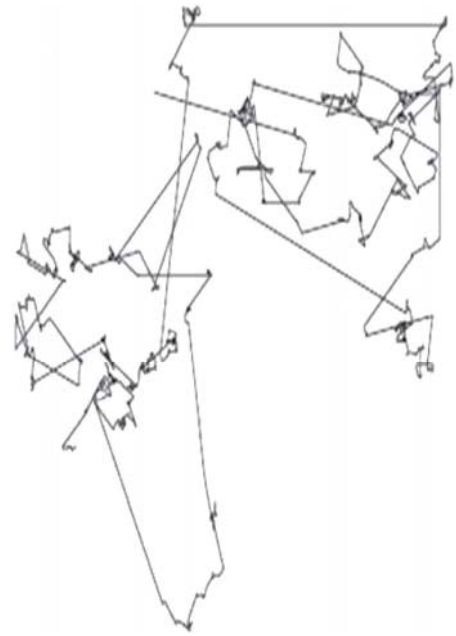


Figure 2: Truncated Levy Walk mobility model's waypoints.

3.2. Observations

3.2.1. For Flight Length Distribution

(i) Power law distribution of flight length implies large number of frequent long flights as compared to Gaussian or an exponential distribution. For humans the duration of moves or the distance moved in one "step" (i.e., flight length) is limited owing to physical constraints. This results in truncated power law distribution for human flight length.

(ii) Heavy tail of flight length implies the declining hazard rate and is captured in terms of conditional expectation [39]

$$E[X/X > k] \propto k. \tag{3}$$

This is also referred as an expectation paradox [45], which means if the observations of heavy-tailed interarrivals are made, then the longer a node has waited, the longer it should expect to wait. This is contrasted with exponential tails where one always gets to the point where the longer one waits, the less time one should expect to continue waiting.

(iii) The tail of the flight length distribution may be long or short and is related to the context of the location. Small and/or highly crowded area encourage shorter flights and discourage longer flights; thus the flight length distribution results in a short-tailed distribution.

(iv) However, with a really large area the truncations will not be excessive. When truncations have less impact on flight lengths, the mobility of the nodes has a stronger power-law tendency [25].

From opportunistic forwarding point of view, node exhibiting large flight length should be chosen as a relay. Based on (3), the concept of "super length parameter" is incorporated in HMSCAR algorithm.

3.2.2. For Pause Time Distribution

(i) Heavy tail of pause time implies the mass-count disparity

$$\lim_{x\to\infty}\frac{P[X_1+\cdots+X_n>x]}{P[\max(X_1+\cdots+X_n)>x]}=1,\quad n\geq 2. \tag{4}$$

This means that when considering collections of observations of a heavy-tailed pause time random variable, the aggregated mass contained in the small observations of pause times is negligible compared to the largest observation in determining the likelihood of large values of the sum. In practice, this means that the majority of mass in a set of observations of pause times is concentrated in a very small subset of the observations. To opportunistically exploit pause time, the node pauses for a longer time duration (corresponds to small subset of the observations) should not be chosen as a relay. Based on (4), a new parameter called "Super pause time" has been incorporated in HMSCAR algorithm to predict node, exhibiting the majority of mass in a set of pause time observations.

(ii) For small or crowded area, the truncation will be short tailed. High traffic will prevent the walker from halting at one location for a long time.

(iii) For really large areas, truncation will be heavy tailed or even long tailed.

The main assumptions made for the system are as follows.

(i) All N nodes are uniformly distributed. The Base Stations (BSs) are located at the edge of the region.

(ii) Nodes can communicate with neighboring nodes, either using cellular bandwidth or short-range communication outlets such as WI-FI or Bluetooth.

(iii) Nodes have finite buffer size and operate on limited power supply.

4. Human-Mobility-Based Sensor Context-Aware Routing (HMSCAR) Algorithm

Traditionally the SCAR algorithm is used for Mobile Sensor Networks (MSN). We have modified the SCAR protocol by adding two human walk context metrics called H_u and P_u. The basic mechanism for relay node selection remains the same. The parameters H_u and P_u are based on our understanding of human walk characteristics as discussed in previous section. The calculation of H_u at each node is done with the help of GPS and is based on the decision whether a given flight length should be considered as a super flight length or not. Similarly the calculation of P_u is based on the decision that whether a recent pause time is the super pause time or not. The definitions of super flight length, super pause time, and corresponding calculation of H_u and P_u are discussed next.

4.1. Flight Length Parameter H_u. The determination of parameter H_u depends on determination of whether a given flight, which a node takes, is a super flight length or not. Hence we will define super flight length next.

4.1.1. Super Flight Length. Given a set U of mobile nodes following the flight length distribution as a truncated power law, then there exists a value ζ for all nodes u of such a subset, that if a flight length is above ζ, then based on (3) all nodes of such a subset again travels that flight length with high probability. Let $S_{f(u)}$ denote the flight length above ζ taken by node u and is defined as super flight length. The actual physical distance corresponds to super flight length is which $S_{\text{len}(u)}$. Then

$$S_{\text{len}(u)}=\alpha' S_{f(u)}, \tag{5}$$

where α' is a constant and is a large percentage of the simulation area. It also depends on geographical conditions and mobility environment but most of the time it is the characteristics of human walk [25]. The $S_{\text{len}(u)}$ will not hold for small simulation areas (approx. 200 meters across) [25]. $S_{\text{count}(u)}$ in (6) is incremented every time, when a node makes a flight of length more than $S_{\text{len}(u)}$. The parameter H_u in (6) is directly proportional to $S_{\text{count}(u)}$

$$\begin{aligned} H_{(u)}&=\frac{1}{1+e^{-2(S_{\text{count}(u)}-1)}},\quad \text{if } S_{\text{count}(u)}>0,\\ H_{(u)}&=0,\quad \text{if } S_{\text{count}(u)}=0. \end{aligned} \tag{6}$$

A plot of parameter H_u is shown in Figure 3. The parameter value rises from initial value 0 to 1 as nodes take super lengths.

4.2. Pause Time Parameter P_u. The determination of parameter P_u depends on determination of whether a given pause is a super pause or not. Hence we will define super pause time next.

4.2.1. Super Pause Time. Given a set U of mobile nodes following the pause time distribution as a truncated power law, then there exists a value ξ for all nodes u of such a subset, that if the pause time is above ξ, then based on (4), that subset of node u pauses much more than ξ. We call all those pause above ξ as super pause $P_{s(u)}$ of node u. The actual physical time corresponding to super pause time is $P_{\text{stime}(u)}$. Then

$$P_{\text{stime}(u)}=\beta' P_{s(u)}, \tag{7}$$

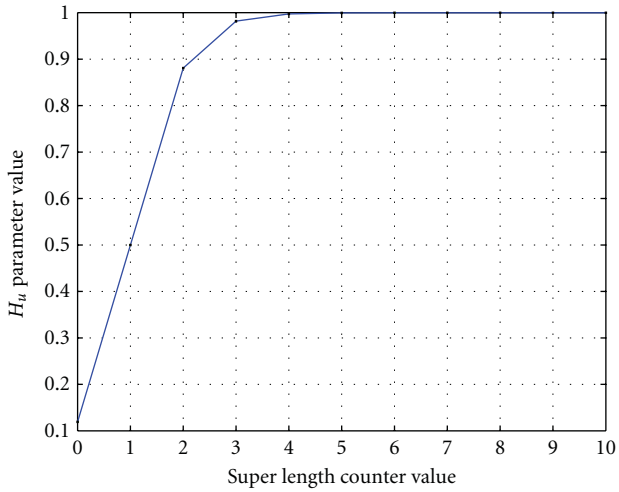


Figure 3: Hu parameter with respect to super length count value.

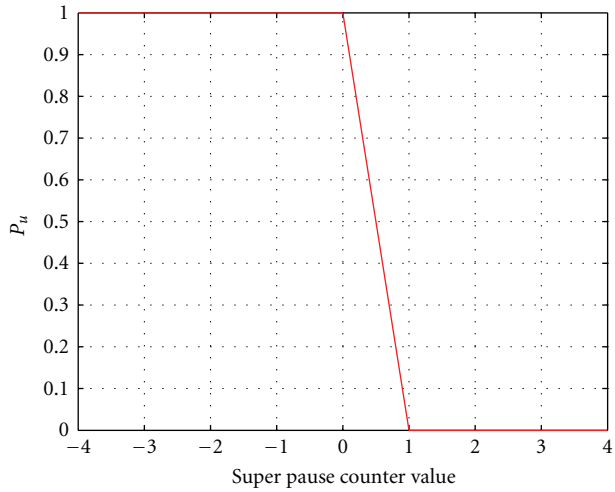


Figure 4: Count function with respect to super pause counter.

where β' is again a constant. Super pause counter $P_{\text{scount}(u)}$ in (8) is changed from 0 to 1 if a node pauses for more than $P_{\text{stime}(u)}$ time.

$$\begin{aligned} P_{(u)} &= 0, \quad \text{if } P_{\text{scount}(u)} > 0, \\ P_{(u)} &= 1, \quad \text{if } P_{\text{scount}(u)} < 0, \end{aligned} \tag{8}$$

A plot of parameter P_u is shown in Figure 4. The difference in (6) and (8) is due to the complementary nature of human mobility context parameters and also due to the fact that the distribution for pause time is less emphatic than that of flight length. The HMSCAR algorithm consists of four different phases and for its implementation clock synchronization among the nodes are required. The different phases of operation are as follows.

4.2.2. Relay Weight Calculation. The calculation of the delivery probability or relay weight is local, and it does not involve any distributed computation. Nodes exchange information about their current delivery probability and available buffer space, periodically with their neighboring node. This also includes routing information.

(1) At the start of data collection round, each node finds all its neighbors (within transmission range R_{tx}). Each node periodically broadcasts NBR_{dis} (Neighbor Discovery) message containing its own ID.

(2) After a certain time T_{boostrap}, each node receives all NBR_{dis} message from its neighbors. Upon receiving NBR_{dis} message from neighbors, each node can estimate the change degree of connectivity by Cdc_u by

$$\text{Cdc}_{(u)} = \frac{|\text{N}u_{t-1} \cup \text{N}u_t| - |\text{N}u_{t-1} \cap \text{N}u_t|}{|\text{N}u_{t-1} \cup \text{N}u_t|}, \tag{9}$$

where $\text{N}u_t$ is the number of nodes in reach of the sensors u at time t and $\text{N}u_{t-1}$ is the number of nodes in reach of the sensors at time $t-1$.

The parameter Cdc_u corresponds to the number of nodes that have transitioned from the in-reach to out-of-reach status or vice versa in the time interval $[t-1, t]$, normalized by dividing it for the total number of nodes met in the same time interval.

(1) Each node calculates the remaining battery parameter Bat_u. The value 1 corresponds to full battery and 0 corresponds to an empty battery.

(2) Each node u summarizes its history of collocation with a sink with parameter Colloc_u where

$$\text{Colloc}_{(u)} = \frac{1}{d}, \tag{10}$$

where d is number of hops from the closest sink. Colloc_u parameter has decreasing value as the distance from a sink increases. If a path does not exist between the sink and the host, its value is set to 0. This information is extracted from the routing table, which is periodically updated depending upon, routing message exchange done with neighboring nodes.

(3) Each node finds H_u as per (6). Each node finds its super length by estimating the distance from received signal strength in some time interval.

(4) Each node finds P_u as per (8).

(5) Each host calculates its delivery probability locally, based on observations related to the various context attributes. Our aim is the optimization of the bundle delivery process. To solve this problem, we apply the so-called weights method [26]. The values of these weights are same for every node.

All the nodes calculate their weight value W_u given by

$$w_u = w_1\text{Cdc}_u + w_2\text{Bat}_u + w_3\text{Colloc}_u + w_4H_u + w_5P_u, \tag{11}$$

where w_1, w_2, w_3, w_4, and w_5 are arbitrarily chosen weighing factors satisfying $\sum_{i=1}^{5} w_i = 1$. The higher the weight value (W_u), the more probable that the node will be a relay node. The BS transmits information about the maximum value of the above parameters attained in the network periodically to all the nodes. All the nodes normalize their parameters with respect to this value.

(1) A timer is used to transmit "WT_{declare}" routing beacon message at a certain time interval, containing the node id, routing table, current delivery probability, and buffer size of the node. This procedure terminates after T_{stop}.

4.2.3. Data Transmission

(9) After T_{stop} time each node receives "WT_{declare}" message from neighboring nodes. Another timer T_1 is used to send data messages periodically. For single copy replication scheme, the compressed data will be sent to the chosen relay node with the highest weight value. If no such neighbor is available, then the message is stored in the buffer and transmitted later. Each message is uniquely identified by the host name and a message number, generated using a counter that is incremented by one for each message sent.

4.2.4. Data Reception

(10) The nodes are also waiting for incoming data. If it is a routing message, then it is processed and corresponding routing table entry is updated. If it is a data message, then an ACK is sent and the message is stored in the buffer. Messages are not deleted from the buffer unless they are acknowledged by some relay node or until the buffer becomes full.

4.2.5. Data Collection Phase at Sinks

(11) Nodes will send their entire data to the sink when they arrive within the transmission range of the sink. Sink aggregates the data packets and sends them to the data repository through a wired network.

5. Performance Metrics

We have evaluated the performance of opportunistic routing algorithm based on received data messages or data delivery ratio and message delay. Moreover, the energy consumption and efficiency of nodes are also evaluated.

(1) Number of Received Packets: it is a measure of the reliability, effectiveness, and efficiency of routing protocols. We mainly consider packet drop only due to nonavailability of end-end path. Other sources of packet drop due to collision, queue overflow, channel unavailability, and so forth are ignored.

(2) Delay: it is the time between packet generation and packet reception at the sink node or the BS.

(3) Energy Consumption: our energy model for the CPSN nodes is based on the first-order radio model described in [46]. A sensor consumes $E_{\text{elec}} = 50$ nJ/bit to run the transmitter or receiver circuitry and $E_{\text{amp}} = 100\,\text{pJ/bit/m}^2$ for the transmitter amplifier. Thus, the energy consumed by a sensor i in receiving a k-bit data packet is given by

$$E_{Rx} = (E_{\text{elec}} \cdot k) \tag{12}$$

while the energy consumed in transmitting a data packet to sensor j is given by

$$E_{Tx} = E_{\text{elec}} \cdot k + E_{\text{amp}} \cdot R_{tx}^2 \cdot k, \tag{13}$$

where "R_{tx}" is the transmission range. One round of operation will consume E_d units of energy due to routing message exchange, where

$$E_d = \sum_{i=1}^{N} (E_{\text{elec}} \cdot k') + \left(E_{\text{amp}} \cdot R_{tx}\right)^2 \cdot k', \tag{14}$$

k' is size of "Hello" packet.

(4) Network Lifetime (Energy Efficiency): the common definitions include the time until the first or the last node in the network depletes its energy. For CPSN, it is defined as the time until an α percent of sensors consume β percent of initial energy, where α is taken as 50 percent and beta is taken as 10 percent.

These parameters are studied for varying buffer size, transmission range, node density, and mobility patterns for single copy replication scheme.

6. Simulation Setup

We measured the performance of our proposed HMSCAR algorithm with SCAR, SFR, and GRAD-MOB on a MATLAB simulator developed to compare the performance of the algorithms. The simulator only simulates the core network characteristics, such as the nodes' connectivity and hello and data message transmissions as these are integral to the understanding of algorithms' behavior. Other details, such as the MAC level protocol, signal propagation, collisions, and realistic channel conditions, are ignored as they have lesser effects. All the algorithms are compared within the same simulation environment making the conclusions unbiased. Further we also assume that there are no obstacles that could limit node mobility or signal propagation.

Due to the importance of connectivity, the weight w_1 associated with Cdc_u was chosen high. The weight values for SCAR algorithm are $w_1 = 0.6$, $w_2 = 0.2$, and $w_3 = 0.2$. The weight values for HMSCAR algorithm are $w_1 = 0.2$, $w_2 = 0.2$, $w_3 = 0.2$, $w_4 = 0.2$, and $w_5 = 0.2$. HMSCAR-sln is a variant of HMSCAR algorithm having weight value 0 associated with the pause time parameter. HMSCAR-sln is also SFR, when gradient scheme according to SCAR is considered. The weight value for HMSCAR-sln or SFR is $w_1 = 0.2$, $w_2 = 0.2$, $w_3 = 0.2$, $w_4 = 0.4$, and $w_5 = 0.0$. While HMSCAR-sps is

Table 1: Default simulation parameters.

Parameters	Values
Sensor field	500 × 500 rectangle area
Number of nodes (N)	150
Transmission range	50
Buffer or queue size	[300, 450, 600, 750]
BS location	(250,250), (0,500), (500,0)
Routing retrans. Interval	60 seconds
Message retrans. Interval	60 seconds
Sampling time	1 minute
Total simulation time	60 minutes
Contextual constant α'	425 meters
Contextual constant β'	20 minutes
Packet header size	25 bytes
Initial energy	2 J/battery
Data packet size	500 bytes
Routing packet size	25 bytes
Scale parameter for flight length	10
Scale parameter for pause time	1

an another variant of HMSCAR, which has weight value 0 associated with flight length parameter. The weight value for HMSCAR-sps is $w_1 = 0.2$, $w_2 = 0.2$, $w_3 = 0.2$, $w_4 = 0.0$, and $w_5 = 0.4$. The GRAD-MOB algorithm is the one which has a different metric, depending upon the decision of whether a node is moving towards the sink or away from the sink. Based on this decision, a value 1 or 0 is assigned, respectively. The first three metric are same as that of SCAR algorithm. The weight value for GRAD-MOB is $w_1 = 0.4$, $w_2 = 0.2$, $w_3 = 0.2$, and $w_4 = 0.2$. These values are used throughout the first four and sixth simulation scenarios. Montecarlo simulations are performed for calculating various performance metrics. We apply SCAR, SFR, GRAD-MOB, HMSCAR, and HMSCAR-sps to the TLW mobility model. Table 1 presents a summary of the already listed simulation parameters. These default parameters are used in all simulations, except where otherwise noted.

(1) Simulation-1: a network of 150 nodes, which move in a rectangular area of 500 m × 500 m, are considered. This is a high density scenario. Simulation was carried out for a total duration of 3600 seconds, and buffer sizes are varied from 300 to 750. Monte Carlo simulations have been carried out for 10 random seeds and 50 meter transmission range. Multicopy replication scheme is implemented, and performance of SCAR, HMSCAR, SFR, and GRAD-MOB is compared.

(2) Simulation-2: simulation scenario similar to part-1 is considered but instead of multicopy, the single copy replication scheme is implemented and performance of SCAR, HMSCAR, SFR, and GRAD-MOB is compared.

(3) Simulation-3: rectangular area of 500 m × 500 m is considered. The node density is varied from 75 to 150 with fix buffer size of 1200. Simulation was carried out for a total duration of 3600 seconds. Monte Carlo simulations have been carried out for 10 random seeds and node communicate with transmission range of 50 meter. Single-copy replication scheme is implemented and performance of SCAR, HMSCAR, SFR, and GRAD-MOB is compared.

(4) Simulation-4: here we vary the transmission range of the nodes from 20 meter to 50 meter for rectangular area of 500 m × 500 m and for 150 nodes having buffer size of 900. The simulation duration was 3600 seconds. Single-copy replication scheme is implemented and performance of SCAR, HMSCAR, SFR, and GRAD-MOB is compared.

(5) Simulation-5: here we vary the weight values of HMSCAR algorithm to see its effect on performance metrics. The transmission range is kept at 50 meter, and 150 nodes moving in a rectangular area of 500 m × 500 m according to TLW mobility pattern are chosen. Buffer size is varied from 300 to 750 for total simulation time of 3600 seconds. Single-copy replication scheme is implemented with six different combinations of weight values. As a first part, the weight values of Cdc_u and P_u are relatively changed while other weight values are not changed. The weight combinations are wt_1 = [0.1 0.2 0.0 0.1 0.6], wt_2 = [0.3 0.2 0.0 0.1 0.4], and wt_3 = [0.5 0.2 0.0 0.1 0.2]. In second part, the weight values of Colloc_u and P_u are relatively changed while other weights are unchanged. These weight combinations are wt_4 = [0.1 0.2 0.1 0.1 0.6], wt_5 = [0.1 0.2 0.3 0.1 0.4], and wt_6 = [0.1 0.2 0.5 0.1 0.2].

(6) Simulation-6: authors of [25] has shown that the realistic human walk traces can be well fitted using TLW mobility model with various combination of α and β. Here we take different combination of β and α for 150 nodes having fixed buffer size of 900 with 3600 seconds of simulation time. Single-copy replication scheme is implemented and performance of SCAR, HMSCAR, SFR, and GRAD-MOB is compared for single-sink and multisink case.

7. Results and Discussions

7.1. Simulation-1. Figure 5 shows number of data packets received for SCAR, HMSCAR, SFR, GRAD-MOB, and HMSCAR-sps algorithms for single-sink and multisink case. The HMSCAR and HMSCAR-sps algorithm performs best among all the five algorithms. Data delivery is improved by 4% compared to SCAR and GRAD-MOB and 3% compared to SFR algorithm for multisink case. Improvement is negligible for single-sink case.

Figure 6 shows the number of packets received for SCAR, HMSCAR, SFR, GRAD-MOB, and HMSCAR-sps algorithms for various buffer sizes. The number of packets received is more for HMSCAR-sps and HMSCAR algorithms compared to other algorithms. HMSCAR has 5% more

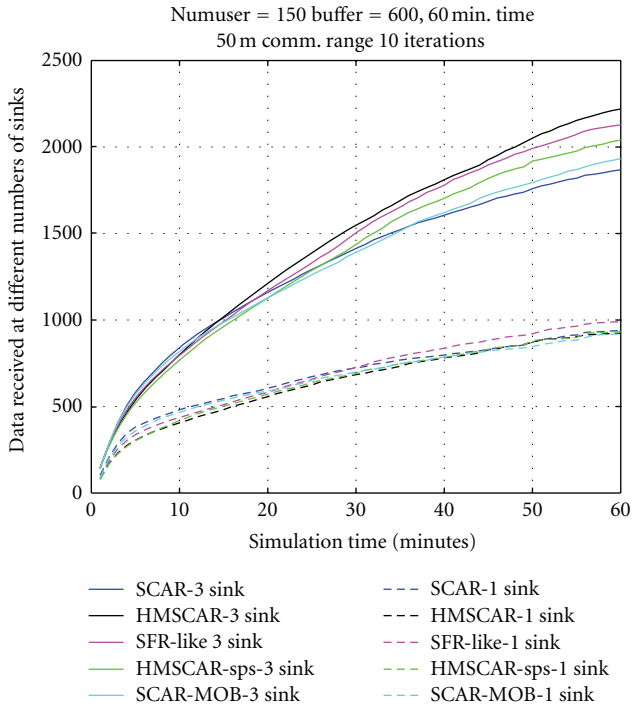


Figure 5: Data received at various sinks versus simulation time.

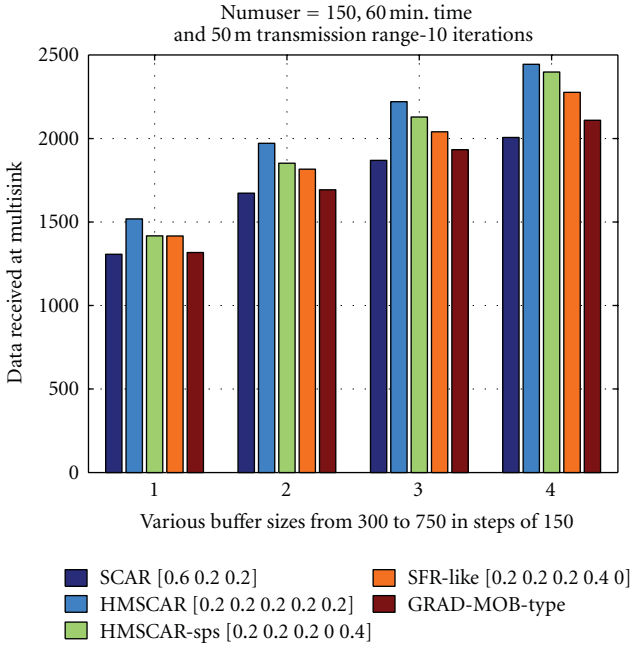


Figure 6: Data received at sinks versus buffer size.

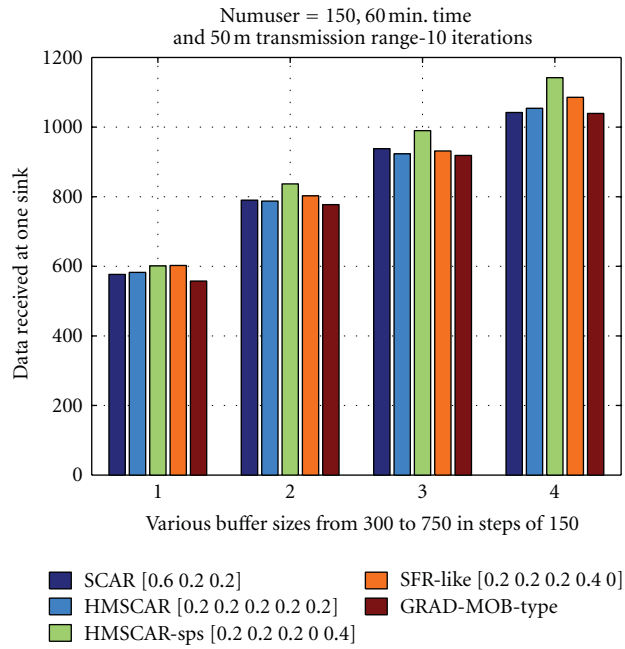


Figure 7: Data received at one sink versus buffer size.

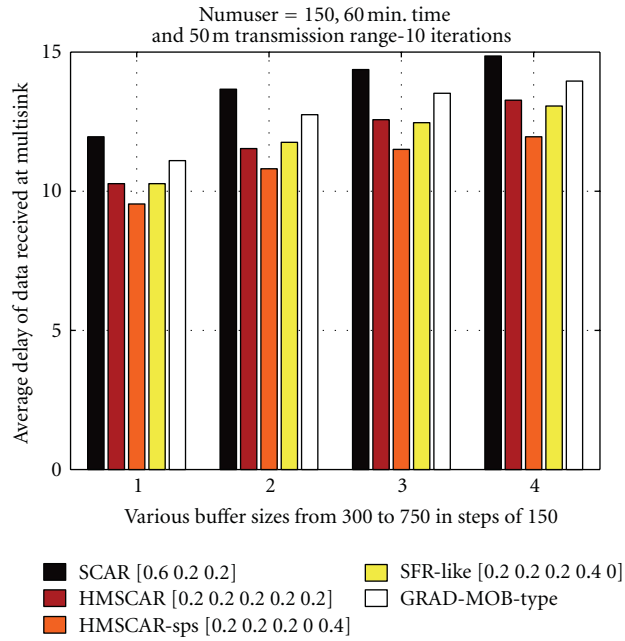


Figure 8: Average delay of packets at multisinks versus buffer size.

message delivery ratio compared to SCAR for buffer size of 750 and has 4% more for buffer size of 600.

Figure 7 shows the number of packets received for various algorithms at single sink. The number of packets received is more for HMSCAR-sps algorithm for all buffer sizes. SCAR and GRAD-MOB have the least amount of received packets.

Figure 8 shows average data delivery delay of the packets for SCAR, HMSCAR, SFR, GRAD-MOB, and HMSCAR-sps algorithms. Average data delivery delay is least for HMSCAR-sps algorithm as compared to other four. SCAR performs worst across various buffer sizes. HMSCAR-sps's performance decreases at higher buffer sizes. This is due to multicopy approach, which stores redundant data in the buffer.

Figure 9 shows average data delivery delay of the packets for SCAR, HMSCAR, HMSCAR-sln, and HMSCAR-sps

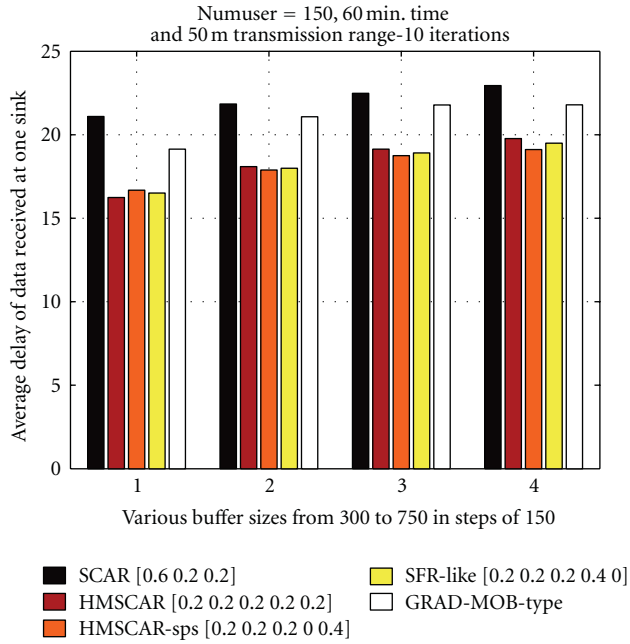


Figure 9: Average delay of packets at single sink versus buffer size.

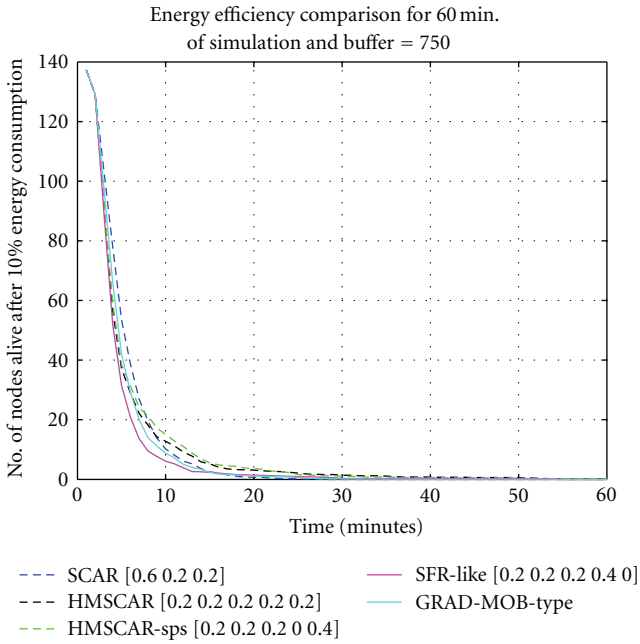


Figure 11: Number of nodes alive after 10% energy consumption.

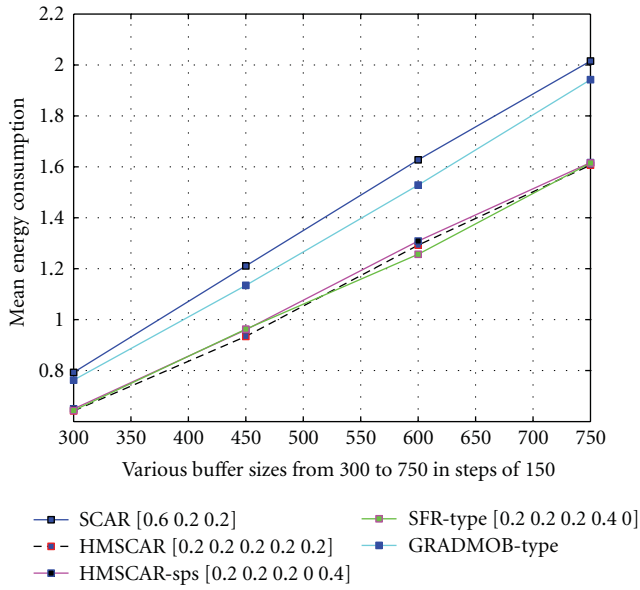


Figure 10: Average energy consumption of nodes versus buffer size.

algorithms. Average data delivery delay is least for HMSCAR, SFR, and HMSCAR-sps algorithms as compared to other two. HMSCAR performs the best across various buffer sizes. SCAR performs the worst among all algorithms.

Figure 10 shows energy consumption of all the nodes for various algorithms. Average energy consumption increases with increase in buffer size. HMSCAR and its variants have minimum energy consumption compared to SCAR and GRAD-MOB.

Figure 11 shows energy efficiency for various algorithms. All the algorithms perform more or less the same. No algorithm has exceptional energy efficiency compared to others. Result shows that HMSCAR performs some what better compared to SCAR and others.

7.2. Simulation-2. Figure 12 shows the number of packets received at the sink for single-copy replication scheme of various algorithms, with different number of sinks. HMSCAR outperforms SCAR almost 22%, and 15% to GRAD-MOB for fixed buffer size of 600.

Figure 13 shows the number of packets received for SCAR, HMSCAR, HMSCAR-sln, and HMSCAR-sps algorithms. The number of packets received is more for HMSCAR algorithm. Performance of all the algorithms increases as buffer size is increased. SCAR has the least amount of received packets amongst all the algorithms.

Figure 14 shows the number of packets received for SCAR, HMSCAR, SFR, GRAD-MOB, and HMSCAR-sps algorithms. The number of packets received is more for HMSCAR-sps algorithm for various buffer sizes. SCAR has least amount of packets received among all the algorithms. Performance of all the algorithms is improved with an increase in the buffer size.

Figure 15 shows average data delivery delay of the packets of various algorithms for single-copy replication case. Average data delivery delay of HMSCAR-sps algorithm is the least. HMSCAR and SFR have poor performance for various buffer sizes. Increasing buffer size also increases data delivery delay. Increasing buffer size enables more messages to be delivered, which were dropped earlier due to insufficient buffer space, but now they are able to reside in the buffer long enough to be delivered to the sink. This incurs a larger delay for these messages and, therefor, increases the average data delivery delay.

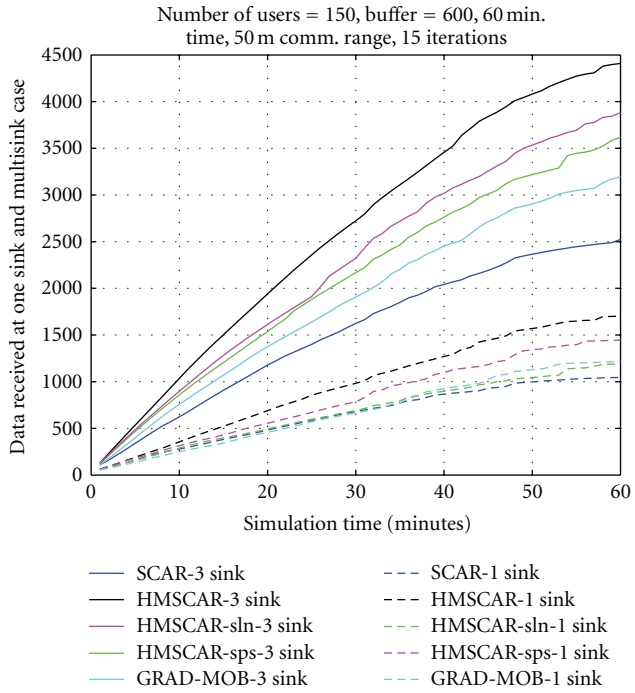


Figure 12: Data received versus simulation time for single copy case.

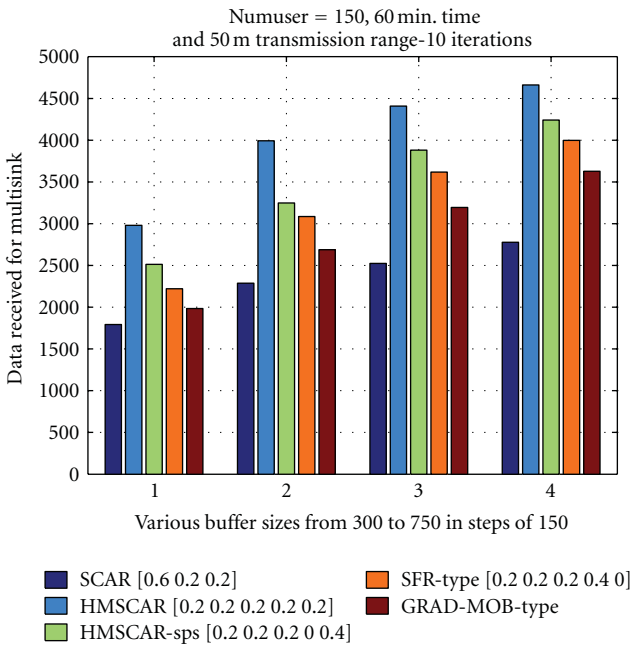


Figure 13: Data received at multisinks versus buffer size.

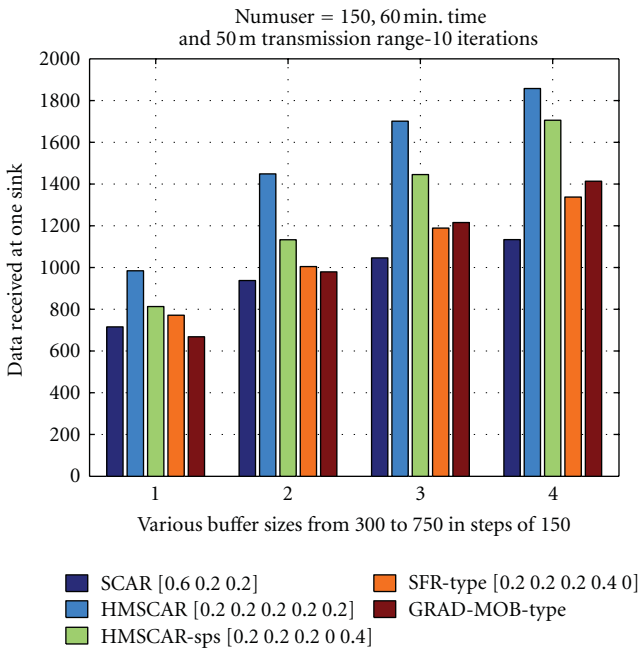


Figure 14: Data received at one of the sink versus buffer size.

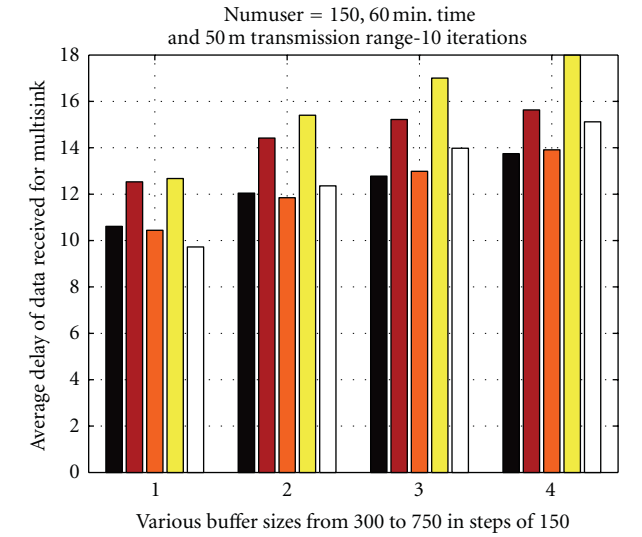


Figure 15: Average delay of packets received for various buffer size.

Figure 16 shows average data delivery delay of the packets for SCAR, HMSCAR, SFR, GRAD-MOB, and HMSCAR-sps algorithms for single-copy, single-sink case. The SFR performs worst and so is HMSCAR. Performance of SCAR is the best for various buffer size.

Figure 17 shows energy consumption of data received at all the sinks for various algorithms. The HMSCAR-sln algorithm consumes more energy while HMSCAR-sps consumes least. For HMSCAR-sln, algorithm the nodes making super flight length are preferred as a relay, so they carry most of the load of the network. While HMSCAR-sps algorithm prevents certain nodes as a relay node and thereby improves data delivery by not carrying network load and therefore its energy consumption is minimum at the expense of data delivery delay.

Figure 18 shows energy efficiency for various algorithms. HMSCAR is energy efficient compared to SCAR and GRAD-MOB. This is obvious as SFR and HMSCAR prefer nodes

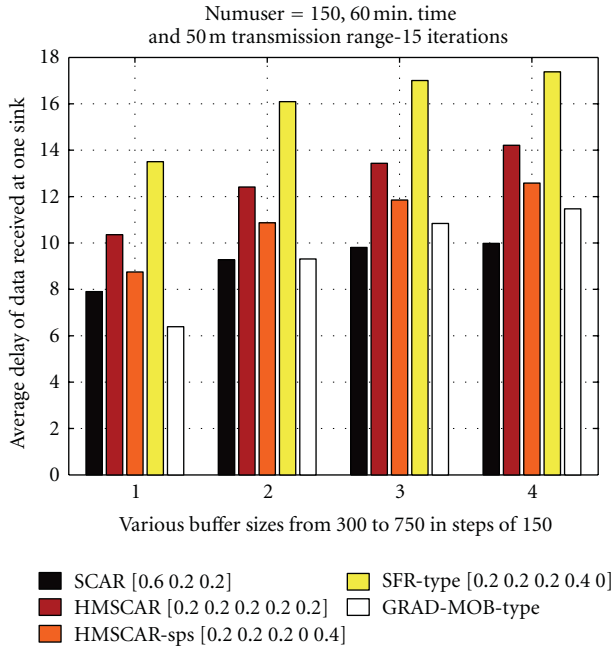


Figure 16: Average delay of packets received for various buffer size.

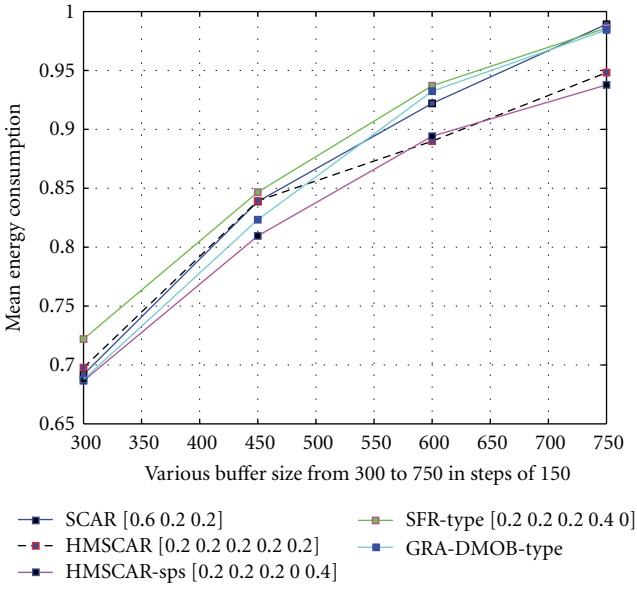


Figure 17: Average energy consumption of nodes for various buffer sizes.

with long jump as relay nodes and most of the load is carried by these nodes, so these nodes will deplete their energy very soon. So instead of 10% node death, if first node death is considered as energy efficiency, then HMSCAR performs the best.

7.3. Simulation-3. Figure 19 shows the number of packets received of various algorithms for various node densities with single-copy replication scheme. HMSCAR has the highest amount of packets received among all the algorithms

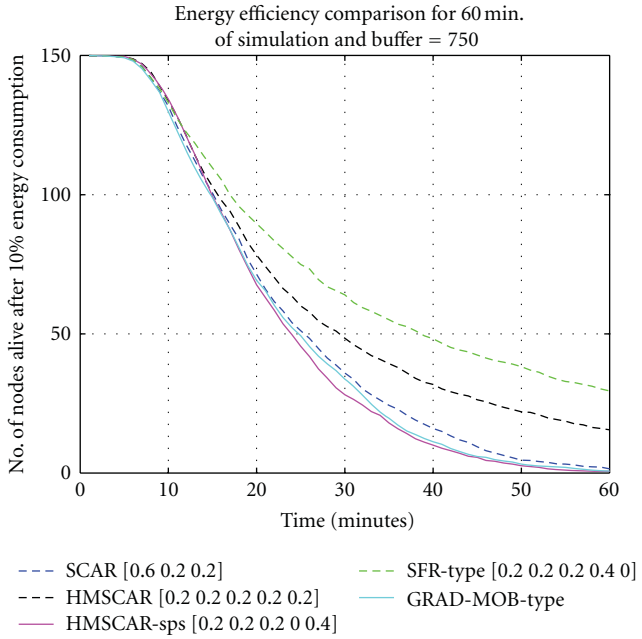


Figure 18: Number of nodes alive after 10% energy consumption.

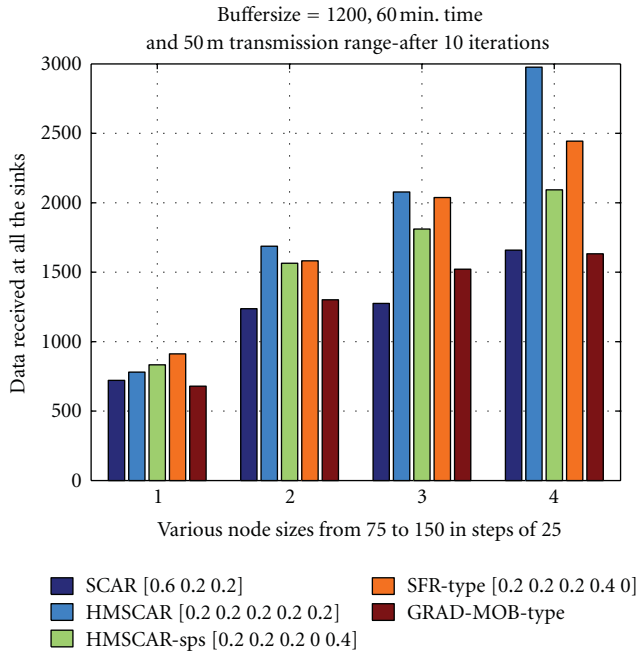


Figure 19: Data received at multisinks versus various number of nodes.

at higher node densities. The number of packets received for HMSCAR is 15% more compared to SCAR for 150 number of nodes and higher by 5% at 100 number of nodes. Performance of all the algorithms improves as node density is increased.

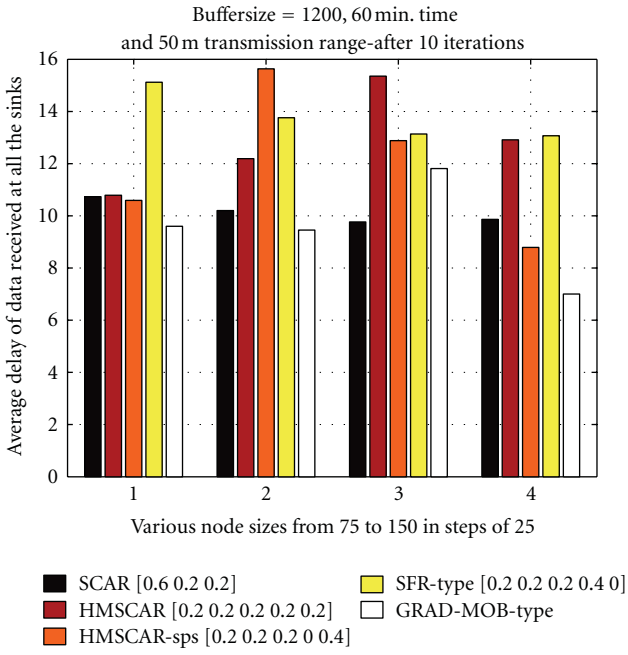


Figure 20: Average delay of packets received for various no. of nodes.

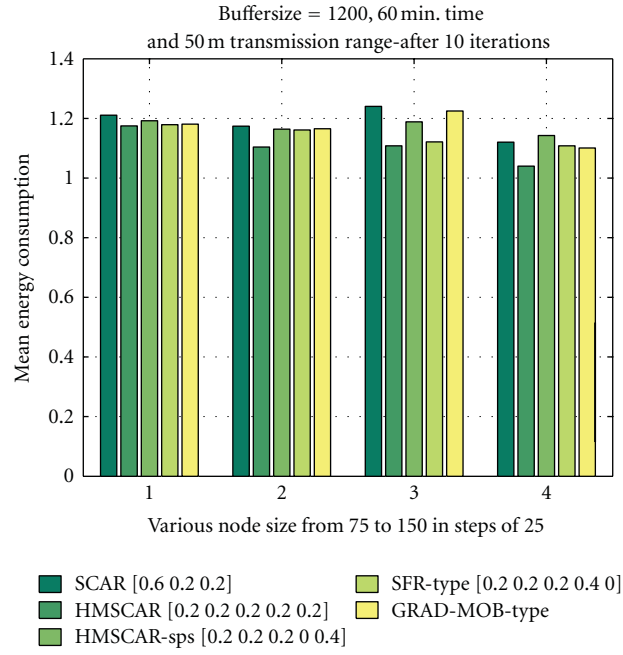


Figure 21: Average energy consumption of nodes for various no. of nodes.

Figure 20 shows average data delivery delay of the packets for SCAR, GRAD-MOB, HMSCAR, SFR, and HMSCAR-sps algorithms with various node densities with the single-copy replication scheme. Average data delivery delay is worst for HMSCAR and its variants at lower node densities. Performance of HMSCAR is improved as node density is increased. It is important to note that, although the HMSCAR performs poor, than GRAD-MOB at node density of 150, the delivery ratio of HMSCAR is 15% more compared to GRAD-MOB at this node density.

Figure 21 shows the average energy consumption of all the nodes for various algorithms. Mean energy consumption is reduced as node density increases. This is due to the decrease in internode distance, that a packet has to travel with relative increase in node density. HMSCAR has least amount of energy consumption among all the algorithms for various node densities.

7.4. Simulation-4. Figure 22 shows the number of packets received of various algorithms for various transmission range with single-copy replication case. The number of packets received is more for HMSCAR-sps algorithm for higher transmission range. HMSCAR-sps has highest packet received amongst all the algorithms for transmission range above 30. At low transmission range, packet delivery is very poor. This is due to unavailability of super nodes as relays at lower transmission range.

Figure 23 shows average data delivery delay of various algorithms for various transmission range with single copy replication case. Average data delivery delay is worst for SFR and HMSCAR algorithms when compared to other three at

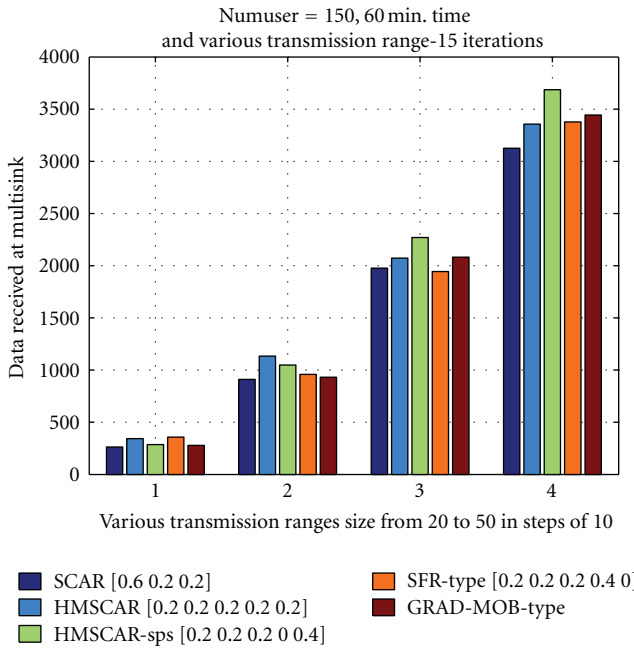


Figure 22: Data received at multi sinks versus various trans. Range.

lower transmission range. Performance is almost equal at higher transmission range.

Figure 24 shows energy consumption of data received at all the sinks for various algorithms. The SFR algorithm consumes more power while HMSCAR-sps consumes the least. Energy consumption increases for change in transmission range from 20 meter to 40 meter. For transmission range of 50 meter, the energy consumption is reduced. Connectivity

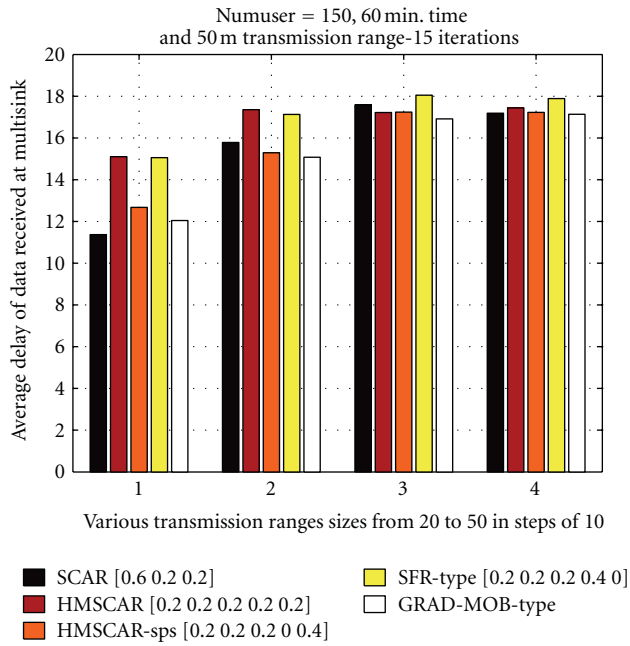


Figure 23: Average delay of packets for various trans. Range.

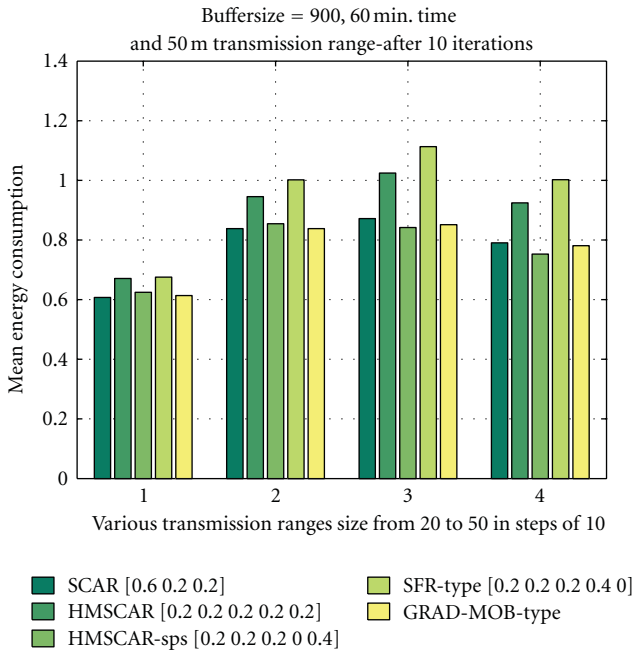


Figure 24: Average energy consumption of nodes for various trans. Range.

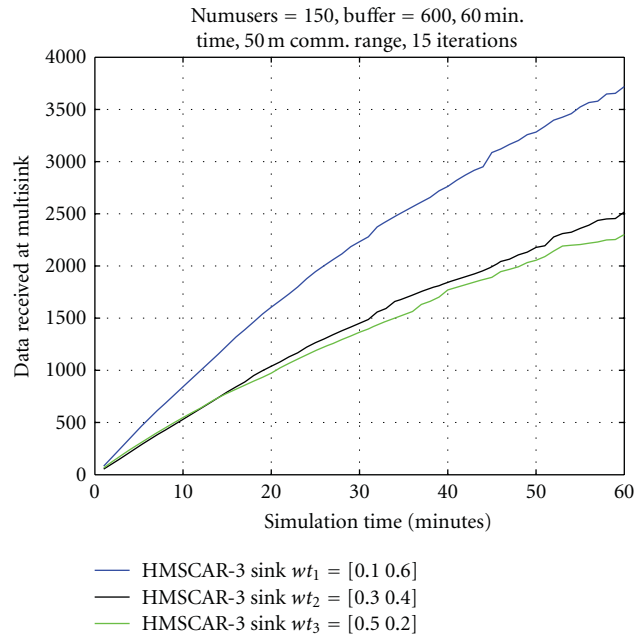


Figure 25: Data received versus simulation time for various w_1 and w_5.

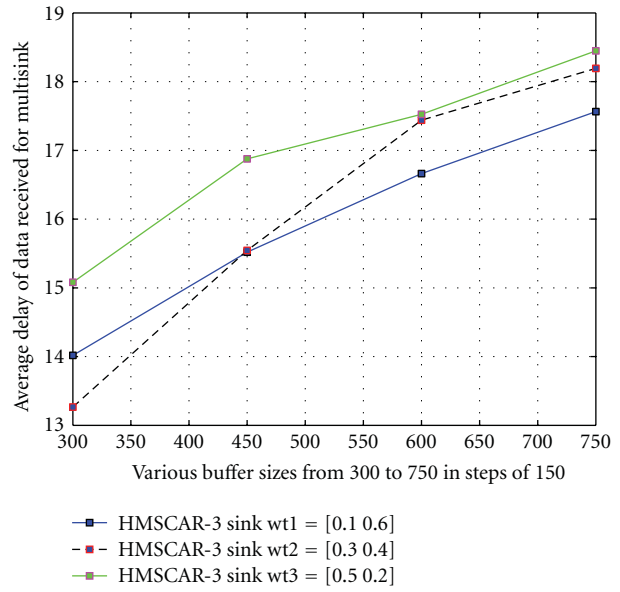


Figure 26: Average delay of packets received for various w_1 and w_5.

with sink node is increased due to increase in transmission range, and therefor the relative internode distance which a packet has to travel from source to destination is decreased.

7.5. Simulation-5. Figure 25 shows the number of packets received for HMSCAR algorithm with various weight combinations of weight w_1 and w_5. The number of packets received is more for higher value of w_5 compared to w_1. This shows the effectiveness of our proposed scheme.

Figure 26 shows average data delivery delay of the packets for HMSCAR algorithm with various weight combinations of weights w_1 and w_5. Average data delivery delay is least for higher value of w_5. Increasing buffer size also increases delay of messages.

Figure 27 shows energy consumption of data received at all the sinks for HMSCAR algorithm with various weight

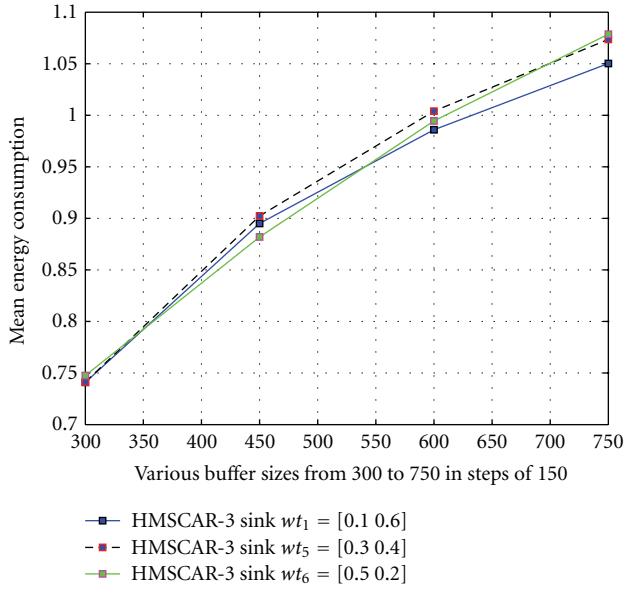


FIGURE 27: Average energy consumption of nodes for various w_1 and w_5.

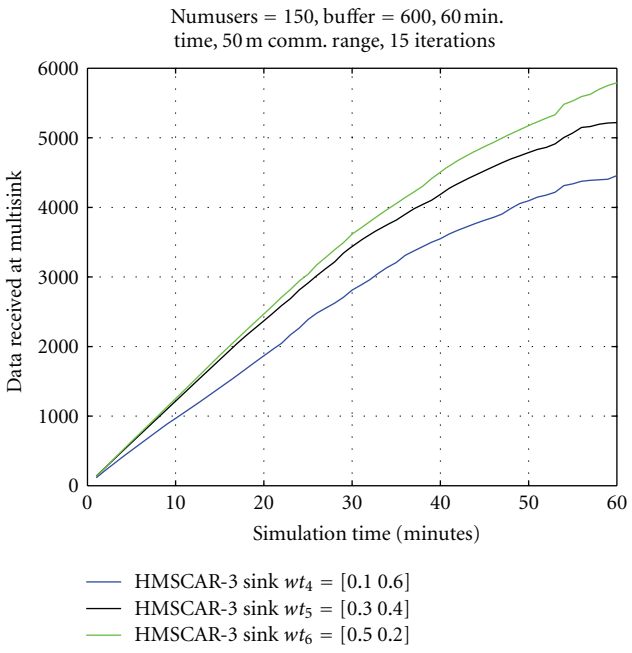


FIGURE 28: Data received versus simulation time for various w_3 and w_5.

combinations of weights w_1 and w_5. The power consumption is almost same for all different combination of w_5 and w_1 for buffer size less than 600. At buffer size 750, the HMSCAR algorithm with higher weight value of w_5 consumes less energy. The energy consumption increases as buffer size increases for all the three different weight combinations.

Figure 28 shows the number of packets received for HMSCAR algorithm with various weight combinations of

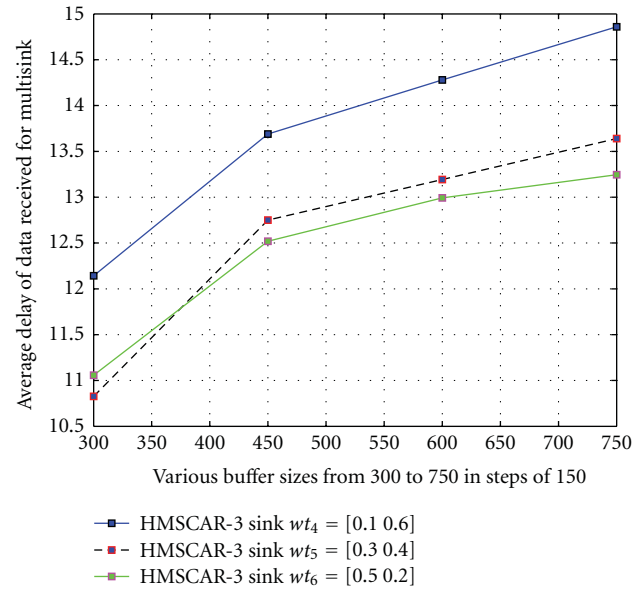


FIGURE 29: Average Delay of packets received for various w_3 and w_5.

weights w_3 and w_5. The number of packets received is more for higher value of weight w_3 compared to w_5. This suggests inefficiency of proposed P_u parameter compared to Colloc_u, when packet delivery ratio is considered as a performance metric.

Figure 29 shows average data delivery delay of the packets for HMSCAR algorithm with various weight combinations of weights w_3 and w_5. Average data delivery delay is worst for higher value of w_5. This suggests the inefficiency of proposed P_u parameter compared to Colloc_u, when packet delay metric is considered. However, it should be noted that calculation of Colloc_u needs routing message exchanges and routing table maintenance at each node, while calculation of P_u does not need message exchanges. It only needs the on-board GPS information, which is easily available on smart phones.

Figure 30 shows energy consumption of data received at all the sinks for HMSCAR algorithm with various weight combinations of weights w_3 and w_5. The power consumption decreases for higher value of w_3 parameter compared to w_5. The energy consumption increases as buffer size increases for all the three different weight combinations.

7.6. Simulation-6. Figure 31 shows the number of packets received for various algorithms with various combinations of levy exponents α and β. For various combinations of α and β, our proposed algorithm outperforms SCAR, SFR and GRAD-MOB. However, it can be observed that the performance of HMSCAR and HMSCAR-sps is best, for smaller value of α which is 0.5. This value corresponds to heavy tail distribution of flight length.

Figure 32 shows the number of packets received of various algorithms with single-sink replication scheme. Result shows that our proposed algorithms HMSCAR and HMSCAR-sps outperform SCAR, SFR, and GRAD-MOB only for α and β equal to [0.5 1.5] and [1, 1], as compared to α

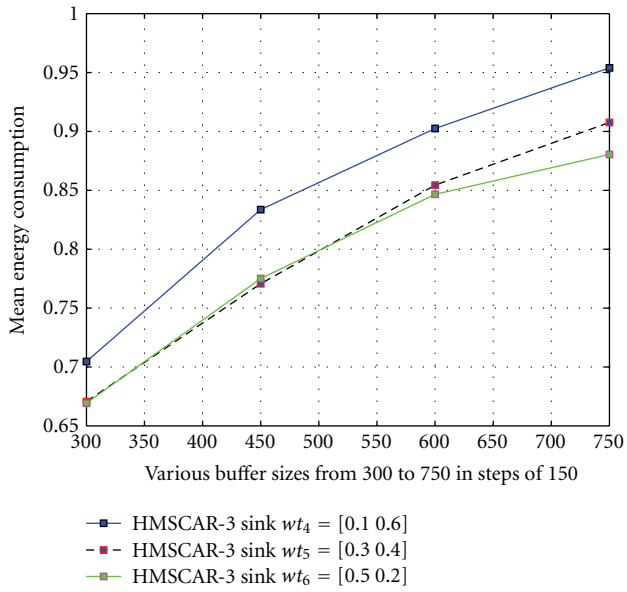


FIGURE 30: Average energy consumption of nodes for various w_3 and w_5.

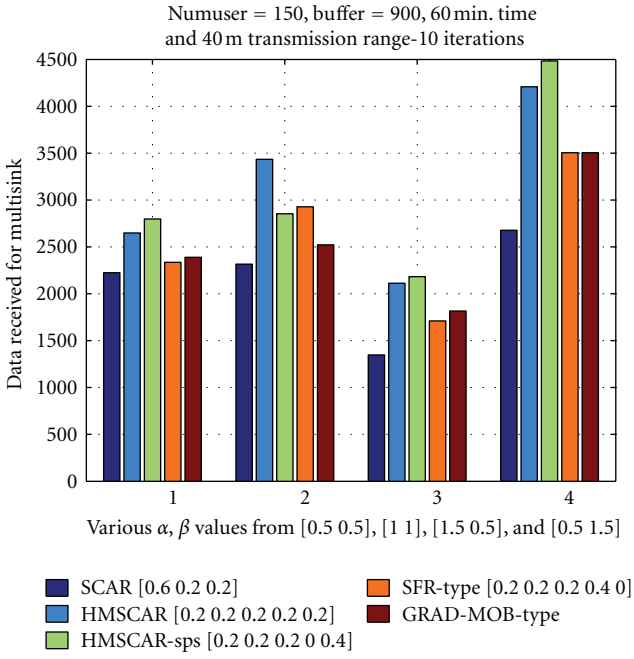


FIGURE 31: Data received versus time for various alpha and beta.

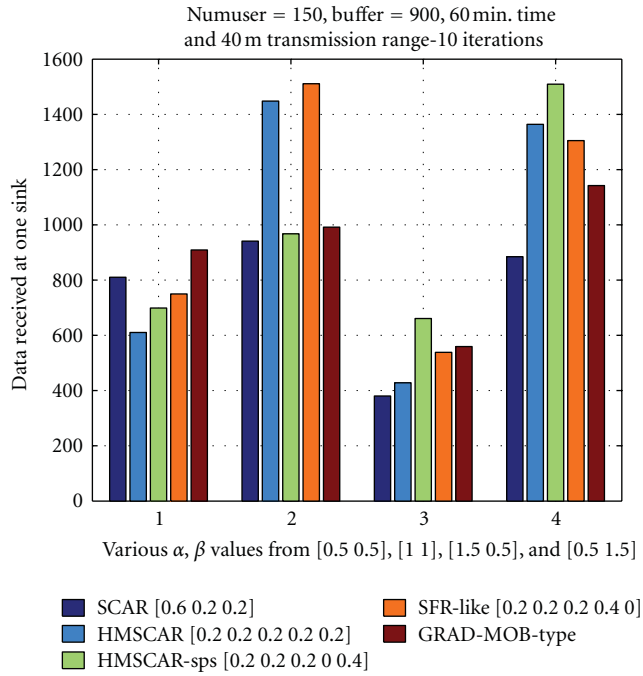


FIGURE 32: Data received versus time for for various alpha and beta.

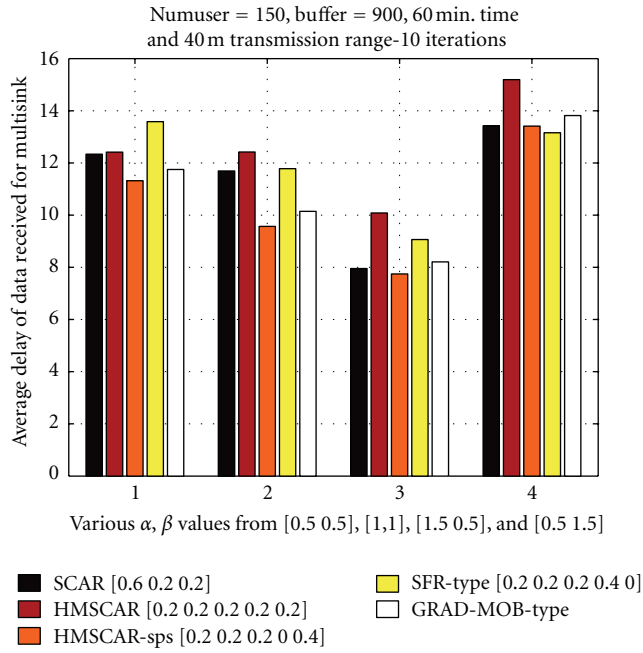


FIGURE 33: Average delay of packets received for various alpha and beta.

and β equal to [1.5 0.5] and [0.5 0.5]. This shows inability of our proposed scheme to perform better compared to SCAR, SFR, and GRAD-MOB in single-sink scenario. The nature of our newly proposed parameters are such that they perform the best for multisink scenario.

Figure 33 shows average data delivery delay of various algorithms with various combinations of levy exponents α and β. Average data delivery delay of HMSCAR-sps is the lowest amongst all the algorithms. This suggests efficiency of proposed P_u parameter for better delay performance under multisink case.

Figure 34 shows average data delivery delay for various algorithms with various combinations of levy exponents α and β. Average data delivery delay of various algorithms is worst for higher value of β equal to 1.5. Average data delivery delay of HMSCAR-sps is 1 the owest among all algorithms

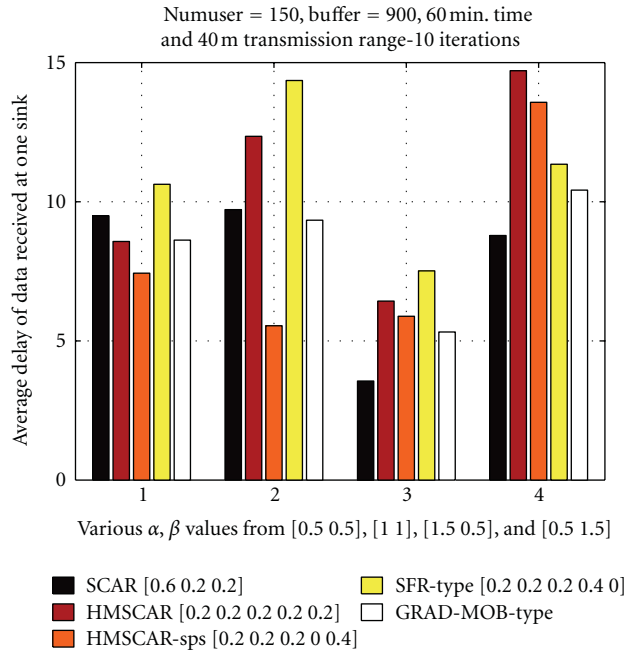


Figure 34: Average delay of packets received for various alpha and beta.

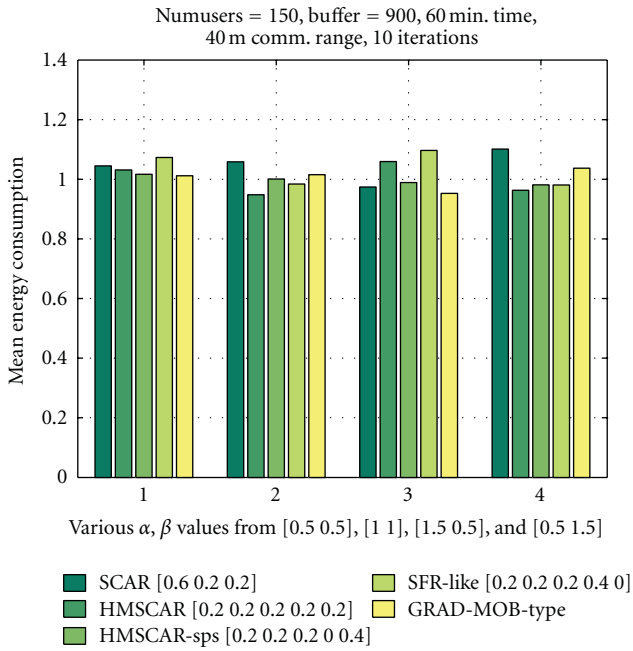


Figure 35: Average energy consumption of nodes for for various alpha and beta.

except for β equals to 1.5. When value of levy walk exponent is near 2, it closely matches Brownian motion. The threshold that is set for deciding super pause time is excessive for this value of levy exponent.

Figure 35 shows energy consumption of data received at all the sinks of various algorithms, with various combinations of levy exponents α and β. Results show that our proposed algorithms, HMSCAR and HMSCAR-sps, outperform the SCAR, SFR, and GRAD-MOB except when α and β equal [1.5 0.5]. The number of super flight lengthes decreases as levy exponent α is increased.

8. Conclusion

This paper presents an opportunistic routing algorithm, which exploits the truncated power law distribution of flight length and pause time of human mobility. The proposed algorithm's performance depends upon availability of nodes, which exhibit super flight length and super pause time. This behavior is observed in tail part of the distribution and is a rare event. The probability of occurrence of this rare events is more for dense networks like CPSN. The majority of DTN routing schemes are aimed towards random mobility and are unrealistic for CPSN scenario. Proposed algorithm considers realistic mobility of cell phone users and performs better with realistic human mobility. The algorithms design is such that it selects nodes with super flight length as relay nodes and avoids nodes likely to observe super pause time, as the relay node. This is quite a natural observation for human mobility. Simulation results indicate HMSCAR's best performance is under 3-sink presence compared to 1. Due to nature of newly introduced utility parameters, it performs better for multi-sink scenario. We have evaluated algorithm for single copy replication scheme, and data delivery and delay are improved simultaneously. This shows the power of newly introduced context parameters, which incorporates human walk characteristics into its design. With multisink presence HMSCAR and HMSCAR-sps perform best for all types of human walk (various combinations of levy exponent α and β). Since single-replication scheme is basis of all other replication schemes, the design of HMSCAR will provide a new insight for routing under human carried mobile devices.

Acknowledgments

This work has been supported by the Department of Science and Technology (DST), Government of India, and the India-UK Advanced Technology Center of Excellence in Next Generation Networks (IUATC) Project.

References

[1] J. Shi, J. Wan, H. Yan, H. Suo, and C. Datong, "A survey of cyber-physical systems," in *Proceedings of the International Conference on Wireless Communications and Signal Processing (WCSP'11)*, pp. 1–6, 2011.

[2] F. J. Wu, Y. F. Kao, and Y. C. Tseng, "From wireless sensor networks towards cyber physical systems," *Pervasive and Mobile Computing*, vol. 7, no. 4, pp. 397–413, 2011.

[3] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 140–150, 2010.

[4] T. Abdelzaher, Y. Anokwa, P. Boda et al., "Mobiscopes for human spaces," *IEEE Pervasive Computing*, vol. 6, no. 2, pp. 20–29, 2007.

[5] S. Eisenman, N. Lane, E. Miluzzo, R. Peterson, G. Ahn, and A. Campbell, "Metrosense project: people-centric sensing at scale," in *Proceedings of the 1st Workshop on World-Sensor-Web (WSW'06)*, Citeseer, Boulder, Colo, USA, 2006.

[6] S. C. Hu, Y. C. Wang, C. Y. Huang, and Y. C. Tseng, "A vehicular wireless sensor network for CO_2 monitoring," in *Proceedings of the IEEE Sensors Conference (SENSORS'09)*, pp. 1498–1501, Christchurch, New Zealand, October 2009.

[7] B. Hull, V. Bychkovsky, Y. Zhang et al., "CarTel: a distributed mobile sensor computing system," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys'06:)*, pp. 125–138, November 2006.

[8] U. Lee, B. Zhou, M. Gerla, E. Magistretti, P. Bellavista, and A. Corradi, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *IEEE Wireless Communications*, vol. 13, no. 5, pp. 52–57, 2006.

[9] P. Mohan, V. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, pp. 323–336, 2008.

[10] D. Chander, B. Jagyasi, U. B. Desai, and S. N. Merchant, "Spatio-temporally adaptive waiting time for cell phone sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2011, Article ID 962476, 21 pages, 2011.

[11] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 606–620, 2007.

[12] M. Conti, S. Giordano, M. May, and A. Passarella, "From opportunistic networks to opportunistic computing," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 126–139, 2010.

[13] N. D. Lane, S. B. Eisenman, M. Musolesi, E. Miluzzo, and A. T. Campbell, "Urban sensing systems: opportunistic or participatory?" in *Proceedings of the 9th Workshop on Mobile Computing Systems and Applications (HotMobile'08)*, pp. 11–16, February 2008.

[14] S. B. Eisenman, N. D. Lane, and A. T. Campbell, "Techniques for improving opportunistic sensor networking performance," *Lecture Notes in Computer Science*, vol. 5067, pp. 157–175, 2008.

[15] T. Liu, C. Sadler, P. Zhang, and M. Martonosi, "Implementing software on resource-constrained mobile sensors: experiences with impala and zebranet," in *Proceedings of the 2nd international Conference on Mobile Systems, Applications, and Services*, pp. 256–269, ACM, 2004.

[16] Y. Wang and H. Wu, "DFT-MSN: the delay/fault-tolerant mobile sensor network for pervasive information gathering," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM'06)*, Barcelona, Spain, April 2006.

[17] H. A. Nguyen and S. Giordano, "Routing in opportunistic networks," *International Journal of Ambient Computing and Intelligence*, vol. 1, no. 3, pp. 19–38, 2009.

[18] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks," *IEEE Communications Magazine*, vol. 44, no. 11, pp. 134–141, 2006.

[19] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 19–20, 2003.

[20] L. Song and D. Kotz, "Evaluating opportunistic routing protocols with large realistic contact traces," in *Proceedings of the 2nd ACM Workshop on Challenged Networks*, pp. 35–42, ACM, 2007.

[21] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proceedings of the ACM SIGCOMM Workshop on Delay-Tolerant Networking*, pp. 252–259, ACM, 2005.

[22] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Tech. Rep. CS-200006, Duke University, 2000.

[23] I. Psaras, L. Wood, and R. Tafazolli, "Delay-/disruption-tolerant networking: state of the art and future challenges," Tech. Rep., University of Surrey, UK, 2010.

[24] T. Spyropoulos, R. N. B. Rais, T. Turletti, K. Obraczka, and A. Vasilakos, "Routing for disruption tolerant networks: taxonomy and design," *Wireless Networks*, vol. 16, no. 8, pp. 2349–2370, 2010.

[25] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong, "On the levy-walk nature of human mobility," *IEEE/ACM Transactions on Networking*, vol. 19, no. 3, pp. 630–643, 2011.

[26] C. Mascolo and M. Musolesi, "SCAR: context-aware adaptive routing in delay tolerant mobile sensor networks," in *Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC'06)*, pp. 533–538, July 2006.

[27] B. Pásztor, M. Musolesi, and C. Mascolo, "Opportunistic mobile sensor data collection with SCAR," in *Proceedings of the IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems (MASS'7)*, pp. 1–12, October 2007.

[28] H. Kanai, Y. Koizumi, H. Ohsaki, and M. Imase, "Gradient-based routing in delay tolerant mobile sensor networks incorporating node mobility," in *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC'12)*, pp. 235–239, Las Vegas, Nev, USA, 2012.

[29] M. Shin, S. Hong, and I. Rhee, "DTN routing strategies using optimal search patterns," in *Proceedings of the 3rd ACM Workshop on Challenged Networks (CHANTS'08)*, pp. 27–32, September 2008.

[30] G. Cugola and M. Migliavacca, "A context and content-based routing protocol for mobile sensor networks," *Wireless Sensor Networks*, vol. 5432, pp. 69–85, 2009.

[31] J. M. Soares, M. Franceschinis, R. M. Rocha, W. Zhang, and M. A. Spirito, "Opportunistic data collection in sparse wireless sensor networks," *Eurasip Journal on Wireless Communications and Networking*, vol. 2011, Article ID 401802, 2011.

[32] L. Guo, R. Beyah, and Y. Li, "SMITE: a stochastic compressive data collection protocol for Mobile Wireless Sensor Networks," in *Proceedings of the IEEE INFOCOM*, pp. 1611–1619, Shanghai, China, April 2011.

[33] M. Keally, G. Zhou, and G. Xing, "Sidewinder: a predictive data forwarding protocol for mobile wireless sensor networks," in *Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'09)*, Rome, Italy, June 2009.

[34] J. Zhu, J. Cao, M. Liu, Y. Zheng, H. Gong, and G. Chen, "A mobility prediction-based adaptive data gathering protocol for delay tolerant mobile sensor network," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'08)*, pp. 730–734, New Orleans, La, USA, December 2008.

[35] R. Jathar and A. Gupta, "Probabilistic routing using contact sequencing in delay tolerant networks," in *Proceedings of the 2nd International Conference on COMmunication Systems and NETworks (COMSNETS'10)*, Bangalore, India, January 2010.

[36] M. C. González, C. A. Hidalgo, and A. L. Barabási, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008.

[37] K. Minkyong, D. Kotz, and K. Songkuk, "Extracting a mobility model from real user traces," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM'06)*, Barcelona, Spain, April 2006.

[38] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad-hoc wireless networks," in *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1360–1369, April 2001.

[39] M. Crovella, "Performance evaluation with heavy tailed distributions," *Computer Performance Evaluation. Modelling Techniques and Tools*, pp. 1–9, 2000.

[40] A. Kansal, M. Goraczko, and F. Zhao, "Building a sensor network of mobile phones," in *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks (IPSN'07)*, pp. 547–548, April 2007.

[41] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 140–150, 2010.

[42] D. Chander, B. Jagyasi, U. B. Desai, and S. N. Merchant, "Layered data aggregation in cell-phone based wireless sensor networks," in *Proceedings of the International Conference on Telecommunications (ICT'08)*, St. Petersburg, Russia, June 2008.

[43] M. Shah, P. Verma, S. Merchant, U. Desai et al., "Leveraging pause time distributions for stable cluster-based data gathering in multihop cell phone-based sensor network," *IETE Technical Review*, vol. 28, no. 6, p. 470, 2011.

[44] M. B. Shah, P. P. Verma, S. N. Merchant, and U. B. Desai, "Human mobility based stable clustering for data aggregation in singlehop cell phone based wireless sensor network," in *Proceedings of the 25th IEEE International Conference on Advanced Information Networking and Applications (AINA'11)*, pp. 427–434, Biopolis, Singapore, March 2011.

[45] B. Mendelbrot, *The Fractal Geometry of Nature*, New York, NY, USA, 1983.

[46] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, 2002.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 317020, 13 pages
doi:10.1155/2012/317020

Research Article

Continuous Remote Monitoring in Hazardous Sites Using Sensor Technologies

Gianfranco Manes,[1] Giovanni Collodi,[1] Rosanna Fusco,[2] Leonardo Gelpi,[2] and Antonio Manes[3]

[1] *University of Florence and The MIDRA Consortium, 50139 Florence, Italy*
[2] *Health, Safety, Environment and Quality Department, Eni S.p.A., 00144 Rome, Italy*
[3] *Netsens s.r.l, Sesto Fiorentino, 50019 Florence, Italy*

Correspondence should be addressed to Gianfranco Manes, gianfranco.manes@unifi.it

Received 12 March 2012; Revised 14 May 2012; Accepted 18 May 2012

Academic Editor: Chih-Yung Chang

The deployment of a distributed point source monitoring system based on wireless sensor networks in an industrial site where dangerous substances are produced, used, and stored is described. Seven essential features, fundamental prerequisites for our estimating emissions method, were identified. The system, consisting of a wireless sensor network (WSN) using photoionisation detectors (PIDs), continuously monitors the volatile organic compound (VOC) concentration at a petrochemical plant on an unprecedented time/space scale. Internet connectivity is provided via TCP/IP over GPRS gateways in real time at a one-minute sampling rate, thus providing plant management and, if necessary, environmental authorities with an unprecedented tool for immediate warning in case critical events happen. The platform is organised into subnetworks, each including a gateway unit wirelessly connected to the WSN nodes. Environmental and process data are forwarded to a remote server and made available to authorized users through a rich user interface that provides data rendering in various formats, in addition to worldwide access to data. Furthermore, this system consists of an easily deployable stand-alone infrastructure with a high degree of scalability and reconfigurability, as well as minimal intrusiveness or obtrusiveness.

1. Introduction

Volatile organic compounds (VOCs) are widely used in industries as solvents or chemical intermediates. Unfortunately, they include components which, if present in the atmosphere, may represent a risk factor for human health. VOCs are also found as contaminants or as byproducts of many processes, that is, in combustion gas stacks and groundwater clean-up systems. Benzene, for example, is highly toxic beyond a time-weighted average (TWA) limit of 0.5 ppm (parts per million), as compared, for instance, with the TWA limit for gasoline which is in the range of 300 ppm. Detection of VOCs at sub-ppm levels is, thus, of paramount importance for human safety and, consequently, critical for industrial hygiene in hazardous environments.

The most commonly used portable field instruments for VOC detection are hand-held photo-ionisation detectors (PIDs), which can be fitted with prefilter tubes for detecting specific gases. The pluses are that PIDs are accurate to sub-ppm levels and measurements are fast, in the range of one or two minutes; for these thus hand-held PIDs are well-suited to field use. However, they have traditionally had two drawbacks: they require skilled personnel and they cannot provide continuous monitoring. Wireless hand-held PIDs have recently become available on the market, thus overcoming these two limitations, but they have a limited battery life, in addition to being relatively costly. This paper describes the implementation and field results of an end-to-end distributed monitoring system using just such VOC detectors, resulting in real-time analyses of gas concentrations in potentially hazardous sites on an unprecedented time/space scale [1].

Wireless sensor networks (WSNs), equipped with various gas sensors, have been actively used for air quality

monitoring since the early 2000s [2–4]. WSNs have the advantage of offering full coverage of the monitored terrain by collecting measurements from redundant portions of the zone. WSNs are thus the ideal instrument for specific and efficient environmental VOC monitoring [5, 6].

This paper describes the implementation of a distributed network for precise VOC monitoring installed in a potentially hazardous environment in Italy. The system consists of a WSN infrastructure with nodes equipped with both microclimatic sensors as well as VOC detectors and fitted with TCP/IP over GPRS gateways which forward the sensor data via Internet to a remote server. A user interface then provides access to the data as well as offering various formats of data rendering. The continuous monitoring, using a unique blended wired/wireless configuration, of benzene emissions from a benzene storage tank is provided as a specific example of the network's usage. A prototype of this system was installed in the eni Polimeri Europa (PEM) chemical plant in Mantova, Italy, where it has been in continuous and unattended operation since April 2011. This pilot site is testing and assessing both the communications and the VOC detection technologies.

To avoid excavations, a stand-alone system, that is, one relying only on autonomous energy and connectivity resources, was designed and installed. In terms of energy requirements, the VOC detectors proved to be by far the greatest energy user, compared to the computational and communication units. So, to ensure a sustainable battery life for these units, efficient power management strategies were studied and implemented; moreover, the WSN elements were equipped with a secondary energy source, consisting of a photovoltaic panel.

This system represents several firsts. One important novelty is the stand-alone unattended long-term operation of communications in a potentially severely hostile environment. The proprietary communication protocols being used will not be discussed in this paper whose purpose is, instead, a general description of the system. Another breakthrough concerns the sensors; the PIDs' continuous power-on operation is made possible by calibration curve linearization and by their sub-ppm detection capability.

2. Emission Estimation Methods

Since estimating diffuse emissions is more difficult and complex than estimating piped emissions (e.g., by stack measurement), we first established what minimum features an ideal method, for licensing and enforcement purposes, must have. The fundamental criteria we pinpointed are:

(i) being inexpensive;

(ii) being suitable for leak detection (all compounds, all locations);

(iii) being suitable for all of the site's equipment and their phases of operation;

(iv) allowing real time estimation;

(v) allowing easy inspection for enforcement;

(vi) allowing depiction of the emissions over any time period;

(vii) allowing for reconfigurability.

A variety of methods for estimating diffuse emissions have been developed. These range from calculation to measurement, point measuring to remote sensing. Some are suited for leak detection, others for estimating annual emissions, and yet others for both of these functions. Below the main currently available methods are described, none of which, however, meets all of the criteria we identified as necessary for the ideal method.

2.1. Distributed Point Sources. The equipment for this method consists of standard air quality measuring devices. In order to cover all potential emission sources it is common practice to monitor several points; furthermore, instead of just fixed measuring points, mobile continuous sensors may be deployed. With the help of a "reverse" atmospheric dispersion model the emissions can be calculated from downwind air quality data combined with meteorological data. This method allows for estimating total emissions, however, it does not cover high plume emissions. Furthermore, the precise location of a leakage is hard to identify with this system.

2.2. Fixed Beam (Open Path) Optical Absorption Method. This method measures the absorption of an electromagnetic beam (IR and UV) by gases present in ambient air, based on the principle that specific gases will absorb light from known parts of the wavelength spectra. Depending on the amount absorbed between the beam source and the detector (coupled to a spectrometer and computer) the amounts of VOCs are calculated. High plume emissions, however, cannot be measured. Moreover, the exact location of a leakage is hard to find with this method as well.

2.3. Differential Absorption LIDAR (DIAL). Optical measuring techniques were further developed in the late nineties to overcome their limitations in pinpointing leakage and in detecting high-altitude leakage sources, resulting in DIAL (differential absorption LIDAR; LIDAR being light detection and ranging). In this system both the infrared laser beam source and the detector are located at the same end of the beam; the detector then picks up the signal from the small amount of light scattered from aerosol droplets or particles in the atmosphere. The main advantages of DIAL over fixed beam methods are that gas concentration is measured at all points along the path and no height limitations exist. Furthermore, it produces 2D and 3D maps of gas concentrations, making it possible to localise the emissions even within large industrial complexes. In other words, DIAL can estimate the total emission flux as well as localising (unexpected) leakage sources; in addition, it covers all potential emission sources (equipment, storage, loading/unloading, waste water system, etc.). However, both its accuracy of localisation as well as its differentiation between different chemical compounds are

limited. Nevertheless, DIAL is an outstanding complement to standard point-by-point leak detection.

2.4. Tracer Gas. The tracer gas method consists of releasing a tracer gas (usually SF6) at different release areas and at various heights above the surface in the factory area and of measuring the VOC and tracer gas concentrations downwind of the factory via either portable syringe-based samplers or portable gas chromatographs. The emission rates of specific hydrocarbons can be estimated from simple flux reading, assuming nearby stationary wind conditions and with no significant atmospheric reactions or deposition of hydrocarbons or other release gases between the leakage points and the sampling points.

We weighed each of the above methods against the criteria our system required and opted for the distributed point system for the PEM Mantova site as it combines reasonable installation and maintenance costs, reconfigurability.

3. Our System Overview

To craft the system, first off suitable locations (both in terms of representativeness and expected impact) were identified along the perimeter of the industrial area, along with several internal sites where hazardous emissions might potentially occur. Owing to the extension and complexity of the Mantova plant, covering some 300 acres and featuring complex metallic infrastructures, it was decided to subdivide the area involved in the piloting into 7 different subareas. Each subarea is covered by a subnetwork consisting of a sink node unit (SNU) equipped with meteorological sensors, for recording wind speed/direction and relative air humidity/temperature (eni 1 to eni 7 in Figure 1). In addition, the eni 2 unit is further equipped with a rain gauge and a solar radiation sensor. An overview of the system deployed at Mantova is shown in Figure 1.

Each SNU is connected to one or more end node units (ENUs) equipped with VOC detectors (see Figure 2 for an example of a configuration), appropriately distributed across the plant's property. This modular approach allows the system to be expanded and/or reconfigured according to the specific monitoring requirements, while providing redundancy in case of failure of one or more SNUs.

The SNUs forward meteorological data, as well as VOC concentration data, to a remote server; as noted above, Internet connectivity is provided via TCP/IP over GPRS using a GSM mobile network. Wireless connectivity uses a UHF-ISM unlicensed band. Electrical power is provided by both primary sources (batteries) and secondary sources (photovoltaic cells), as mentioned above.

VOC concentration and weather/climatic data are updated every minute. This intensive sampling interval allows the evolution of gas concentrations to be accurately assessed. Furthermore, when all of the weather-climatic measurements are collated, they provide a map of the area's relative air humidity/temperature and wind speed/direction, which are crucial for providing accurate VOC-sensor read-out compensation [7]. The need for so many wind stations across the plant property is warranted by the turbulent wind distribution in that particular area, as can be observed by the different orientations of the blue arrows representing wind direction in Figure 1.

Three of the ENUs—eni 1, eni 2, and eni 3—were deployed along the perimeter of the plant to locally monitor VOC concentration while correlating it with wind speed and direction; the other seven were placed around the chemical plant and in close proximity of the pipeline, which are possible sources of VOC emissions. In fact, potential sources of VOC emissions in the plant are in easily identified areas, such as the chemical plant and the benzene tanks. The chemical plant was surrounded with a high number (6) of VOC sensors, thus creating a virtual fence, capable of effectively evaluating VOC emission sources based on the concentration patterns detected around the plant.

Figure 2 shows the layout of two of the subnetworks, one deployed around the chemical plant and one near the pipeline. The subnetwork around the chemical plant, Figure 2(a), consists of two SNUs, eni 6 and eni 7, equipped with weather sensors (air/wind), each connected with three ENUs spaced tens of meters from each other. The subnetwork located in the pipeline area is shown in Figure 2(b). One of the two ENUs is located in close proximity to the end of the pipeline itself (nodo *4*), while the other (nodo *2*) is a bit further away. Sampling the VOC concentration at intervals of tens of meters allows the dispersion of VOC emissions to be evaluated; in addition, information about wind speed/direction allows the emission's source to be identified.

4. The VOC Detector

The VOC detector is a key element for the monitoring system's functionality. For this application two criteria were considered mandatory. The first is that the VOC detector must be operated in diffusion mode, thereby avoiding pumps or microfluid devices which would increase the energy requirements and make maintainability issues more critical. The second criterion was that the system should be able to operate in the very low part per billion (ppb) range, with a minimum detectable level (MDL) of some 2.5 ppb with a ±5% accuracy in the 2.5 to 1000 ppb range, which represents the range of expected VOC concentrations. Another requirement was operating at one-minute intervals.

The PID fulfills most of the above requirements. Two major issues were identified, however, which potentially impact efficient use of the PID in our system. The first was that in the low ppb range the calibration curve of the PID shows a marked nonlinearity; this would require an individualized meticulous multipoint calibration involving higher costs and complexity. The second issue was that, when operated in free diffusion mode at low ppb and after a certain time in power-off, the detector required a stabilisation time of dozens of minutes, hence it would not be able to operate at our required one-minute intervals.

Since both of the above-mentioned limitations are intrinsically related to the PID's physical behaviour, this

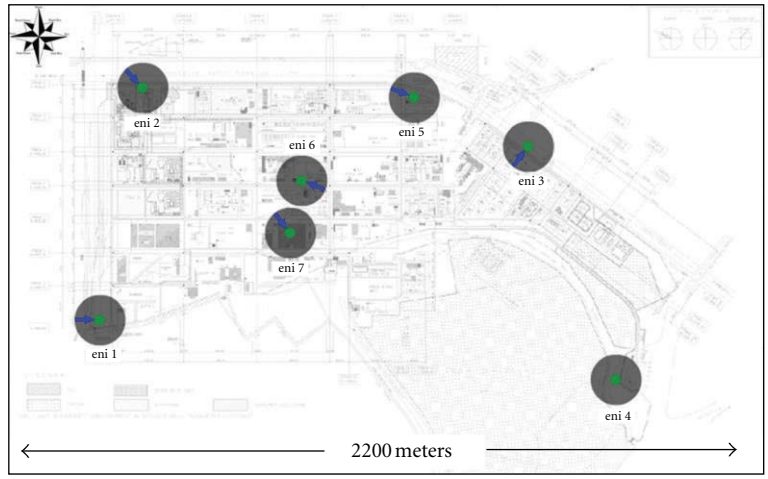
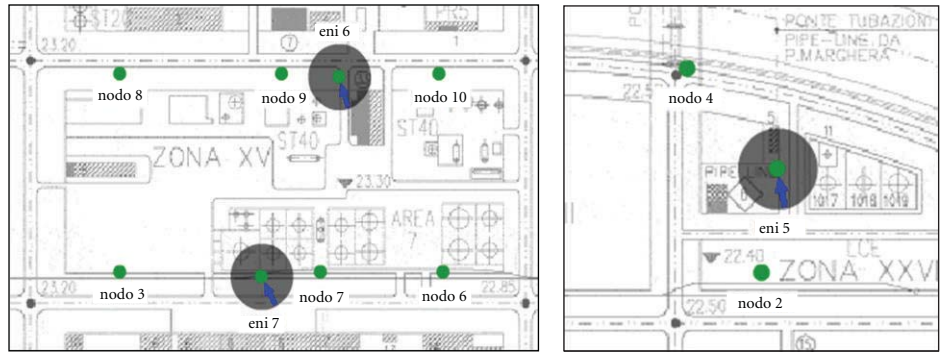


Figure 1: Installation overview. The grey circles indicate the position of each SNU; the blue arrows show wind direction.



(a) (b)

Figure 2: Closeups of SNU and ENU deployment around one of the chemical plants (a) and the pipeline (b). Maps are oriented according to the plant's axes rather than cardinal directions.

was carefully investigated and a behavioural model of the PID was developed to explain these phenomena. To resolve the nonlinearity, a mathematical expression of the PID calibration curve was derived [8]. Accordingly, the PID calibration procedure was adjusted to measure only two parameters: the zero gas voltage and the detector sensitivity in mV/ppm.

The second problem was the stabilisation time required to achieve a stable read-out time in diffusion mode at low concentrations. Originally our system requirements called for a minimum VOC data sampling interval of at least fifteen minutes. This would have had the additional advantage of prolonging the PID's battery life and maintenance, keeping costs down, However, discontinuously operating the PID results in stabilisation times of dozens of minutes in order to get a reliable PID read-out, which conflicts with the system requirement of sampling VOC concentrations in real time. In order to read very low VOC concentration levels in diffusion mode, however, the PIDs have to be continuously powered-on, consuming about 35 mA. Comparing the advantages of the two operation modes, the benefits of discontinuous operation were marginal compared to the matchless advantage of the more time-intensive monitoring of VOC concentration provided by continuous power-on operation. Accordingly, it was decided to operate the VOC detectors continuously at one-minute data sampling, which would also meet our established criteria of real-time estimation. Furthermore, this decision proved to be very effective as some emission events at the plant show very rapid variation, which would be difficult to interpret based on ten-minute sampling rates.

5. Continuous Emission Monitoring at Benzene Storage Tanks

Storage tanks represent an important potential source of VOC emissions and probably account for a significant amount of the site's total diffuse emissions. Thus they need to be appropriately monitored. Emissions from tanks can vary significantly from tank to tank, according to their size, design, maintenance, liquid level, and properties, as well as whether the tank is filling, stable, or emptying. Wind speed can also have a substantial effect on tank emissions, particularly for floating roof tanks.

Benzene storage tanks on this site are of the floating roof type and are located in highly hazardous areas. In fact, the electrical equipment operating in those areas need a special safety certification which classifies each area according to European Directive 94/9/EC (referred to as ATEX, an acronym for the French "Atmosphères Explosibles"), regarding equipment and protective systems intended for use in potentially explosive atmospheres.

The PEM site's benzene storage tank is certified as ATEX Zone 0, which means it is an area where an explosive atmosphere is continuously present or present for long periods of time; hence, the sensors that are located in close proximity to benzene sources must meet ATEX Zone 0 requirements.

The layout of the benzene storage tank monitoring network (STMN) is displayed in Figure 3. The STMN consists of three VOC units, each equipped with a PID and a computational unit, serially interconnected by wires as well as connected to the wireless unit (WU), which provides power and wireless connectivity. The WU is then connected to the GPRS unit (eni 3 in Figure 1) which provides the Internet connectivity. The reason for choosing such a hybrid wired/wireless configuration is due to the VOC detector's energy needs.

As can be observed in Figure 3, the three VOC detectors are located in Zone 0, which requires a very high level of protection, while the WU, along with the power unit consisting of the battery and the photovoltaic panel, needed to meet the VOC detectors' high energy consumption, is located in the non-Zone 0 area. In fact, the VOC unit's current absorption of 35 mA calls for a primary energy source with at least 80 A h capacity for 60 days of continuous operation. This would mean replacing 3 batteries on the top of the storage tank, requiring skilled personnel, every two months. This was considered impractical and too costly. On the other hand, the option of equipping the unit with a secondary energy source, such as a photovoltaic panel, to prolong battery life, was dismissed as incompatible with the safety requirements of an ATEX Zone 0.

As a result, the hybrid configuration of Figure 3 was drawn up, placing the VOC units within Zone 0, yet the communication/power supply units outside of the hazardous area. This permits usage of a secondary energy source, while, at the same time, allowing easy replacement of the primary energy source.

6. The Communications Platform

The communications platform, described more in depth elsewhere [1], is able to support a scattered system of units collecting VOC emission data in real time, while offering a high degree of flexibility and scalability, so as to allow for adding other monitoring stations as needed. Furthermore, it provides reconfigurability, in terms of data acquisition strategies, while being more economically advantageous than traditional fixed monitoring stations.

A GSM mobile network solution featuring a proprietary TCP/IP protocol with DHCP provides Internet connectivity. Dynamic reconnectivity strategies provide efficient and reliable communication with the GSM base station. All the main communication parameters, such as IP address, IP port (server's and client's), APN, PIN code, and logic ID, can be remotely controlled. Wireless connectivity between SNUs and ENUs is performed in an unlicensed ISM UHF band (868 MHz).

6.1. The Sink Node and Wireless Interface Units. A block diagram of the SNU is represented in Figure 4(a). It consists of a GPRS antenna and a GPRS/EDGE quadriband modem, a sensor board, an I/O interface unit, and an ARM-9 microcontroller operating at 96 MHz. The system is based on an embedded architecture with a high degree of integration among the different subsystems. The unit is equipped with various interfaces, including LAN/Ethernet (IEEE 802.1) with TCP/IP protocols, USB ports, and RS485/RS422 standard interfaces. The sensor board has 8 analogue inputs and 2 digital inputs. The SNU is also equipped with a wireless interface (WI), shown in Figure 3(b), which provides connectivity with the ENUs.

The wireless interface (WI), Figure 4(b), provides short-range connectivity. It operates on a low-power, ISM UHF unlicensed band (868 MHz) with FSK modulation; moreover, it features proprietary hardware and communication protocols. Distinctive features of the unit are the integrated antenna, which is enclosed in the box for improved ruggedness, as well as a PA + LNA for a boosted link budget. The PA delivers 17 dB m to the antenna, while the receiver's noise figure was reduced to 3.5 dB, compared with the intrinsic 15 dB NF of the integrated transceiver. As a matter of fact, a connectivity range in line-of-sight in excess of 500 meters was obtained, with a reliable communication with a low BER, even in hostile EM environments.

The energy required for the unit's operation is provided by an 80 A h primary source and by a photovoltaic panel equipped with an intelligent voltage regulator. Owing to its prudent low-power design, the unit can be powered with a small (20 W) photovoltaic panel while maintaining continuous unattended operation.

Given the distance among the sink nodes and the hostile EM environment, implementing a multihop WSN would have been impractical, costly and troublesome in terms of quality of service; our solution, instead, has resulted in practically flawless performance, in terms of reliability, with only a very marginal extra cost.

6.2. The EN Unit. A block diagram of the ENU is shown in Figure 5; it consists of a WI, similar to that previously described, yet includes a VOC sensor board as well as a VOC detector. The acquisition/communication subsystem of the ENU is based on an ARM Cortex-M3 32-bit microcontroller, operating at 72 MHz, which provides the necessary computational capability on the limited power budget available.

To reduce the power requirement of the overall ENU subsystem, two different power supplies have been implemented: one for the microcontroller and one for the peripheral units. The microcontroller is able to connect/disconnect the

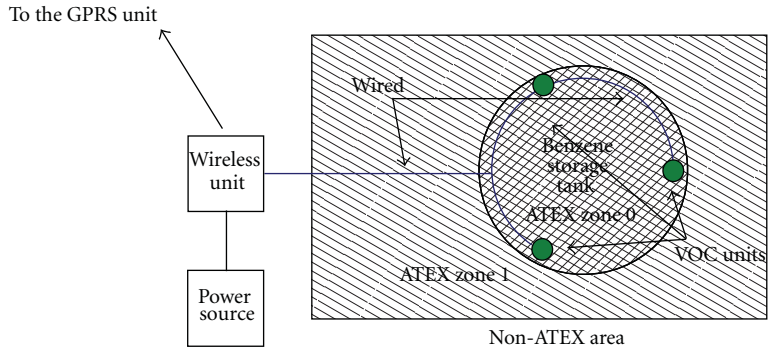
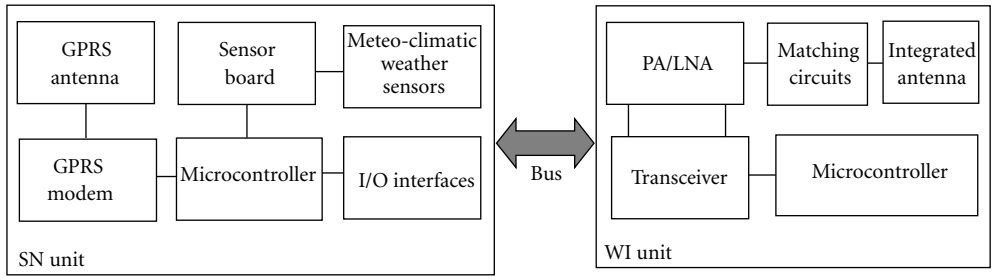


Figure 3: Scheme of the storage tank monitoring network (STMN).



(a) (b)

Figure 4: Block diagram of an SNU (a) and a WI unit (b).

peripheral units, thus preserving the local energy resources. The VOC detector subsystem in particular is powered by a dedicated switching voltage regulator; this provides a very stable and spike-free energy source, as required for proper operation of the VOC detector itself.

Communication between the ENU and the VOC detector board is based on an RS485 serial interface, providing high-level immunity to interference as well as bidirectional communication capability, which is needed for remote configuration/reconfiguration of the unit.

7. WSN Issues

7.1. Network Structure and Routing Schemes. Among the different alternatives, a hierarchical-based routing scheme was selected based on the particular nature of the installation: the extended area of the plant, the few critical areas of potential sources of emissions requiring dense deployment, and the highly uneven distribution of nodes over the area. A hierarchical-based routing scheme fit the projected deployment layout well. As said before, the installation was partitioned into subnetworks to be deployed around the critical sites, with one SNU for each individual subnetwork. In principle, wireless connectivity between the SNUs could have been implemented, using one specific SNU as a gateway to the Internet. This option, however, conflicted with at least two of the major requirements. The first is the need for redundancy in case of failure of the gateway unit; in this scenario, in fact, Internet connectivity would be lost, with consequent loss of the real-time updating capability, which is considered a mandatory requirement for the system. The second need which would not have been met is that of providing full connectivity among the individual SNUs under conditions where line-of-sight propagation was not guaranteed, due to the presence of such temporary obstacles as trucks or maintenance infrastructures. A multiple-GPRS gateway approach overcomes those limitations; even in the case of failure of one or more gateway units, Internet connectivity would be provided by the others still in operation, while the issue of the obstacles is circumvented. As for the wireless connectivity, a star configuration was preferred to a mesh configuration, given the limited number of nodes and the need to keep latency at a minimum.

7.2. Protocols and WSN Services. Two levels of communication protocols, in a mesh network topology, were implemented. The upper level handles communications between the SNs and the server; it uses a custom binary protocol on top of a TCP layer. This level was designed and calibrated for real-time bidirectional data exchange, where periodic signaling messages are sent from both sides. Since our sensor network necessitates a stable link, quick reconnection procedures for whenever broken links should occur, were especially important. To ensure minimal data loss, the SNs have nonvolatile data storage, as well as automatic data packet retransmission (with timestamps) after temporary downlink events. Furthermore, this design is well suited for low-power embedded platforms like ours, where limited memory and power resources are available.

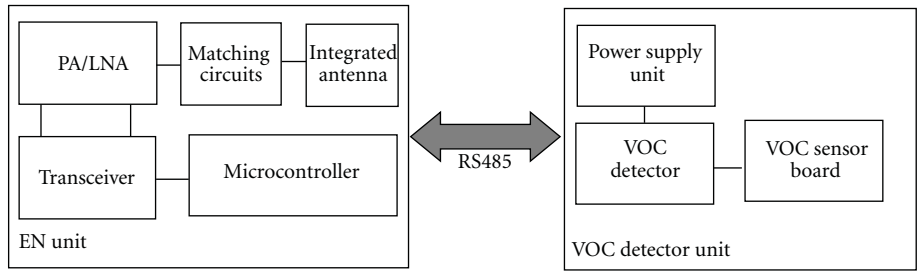


Figure 5: Block diagram of the ENU (a) and the VOC detector unit (b).

In fact, our protocol stack currently requires about 24 KB of flash memory (firmware) and 8 KB RAM.

The lower level, in contrast to the upper one, concerns the local data exchange between the network nodes. Here a cluster tree topology was employed; each node, which both transmits and receives data packets, is able to forward packets from the surrounding nodes when needed. In this specific application, the topology and routing schemes are based on an ID assigned to each EN unit, where the ID can be easily adjusted using selectors on the hardware board. This choice allows for easy support and maintenance, even when non-specialized operators have to install, reinstall, or serve one or more units.

8. Energy Budget Issues

Energy budget plays a key role in the maintainability of the WSN [9]. In our case this is made even more critical by the necessity for stand-alone operation, as well as due to periodic maintenance intervals in excess of four months. Since electrical energy from the plant could not be used, secondary sources had to be locally available; photovoltaic panels (PVP) fit the bill. The SNUs are almost all equipped with PVPs, as they have to support a number of functions, including connectivity and data collection from sensors. The ENUs, when equipped with low-energy sensors, have 3 to 5 years of battery life using primary sources [10]. However, in this system the ENUs have to support the power-hungry VOC sensors as well. For this reason, the ENUs are also equipped with PVPs.

8.1. ENU Energy Budget. The EN nodes have been fully deployed since July 2011; since that time we have noticed that the VOC sensors' energy budget predominates over that of the computational/communication unit. Since the PIDs used for reading the VOC concentration need to be continuously on to operate efficiently, this corresponds to a current draw of some 35 mA, corresponding to 840 mA h a day, more than twice the amount used by the communication/computational units, which consume some 360 mW a day. The ENU's primary source capacity is 60 A h, which provides more than 2 full months of continuous operation.

To enable the ENUs to rely fully on autonomous energy resources while providing continuous operation, a 5 W photovoltaic panel was integrated into the unit in order to supply the additional 360 mW average required power. The PVP, however, can fulfil the task only under ideal sunlight conditions, that is, in summer, but hardly at all in winter. Hence, the PVP power supply unit also has a charge regulator which was specifically designed to provide maximum energy transfer efficiency from the panel to the battery regardless of weather conditions. In conclusion, the secondary energy source plays a key role in ensuring the stand-alone and unattended operation of the communication platform.

As a concrete example, Figure 6 shows the battery voltage plots for the ENUs connected to SNUs 5 and 6. As can be observed, the ENUs exhibit quite satisfactory charge conditions, although ENU 4 (eni 5 nodo 4) shows a slightly lower voltage level, probably due to deployment in a partially shadowed area. The battery voltage remains above a 12.3 V value, with a slightly decreasing trend, possibly due to lower solar energy given the fact that the time period corresponds to the onset of autumn.

8.2. SNU Energy Budget. The SNUs were deployed at the PEM plant in the middle of April 2011. They have much higher energy requirements than the ENUs as they have to supply energy for both wireless connectivity and sensor operation (including meteorological sensor).

The average current draw is around 90 mA, corresponding to a power consumption of about 1 W. SNUs have superior primary and secondary resource capabilities, with a 2-month battery life relying on the primary source alone.

Figure 7 shows the battery voltage for the eni 2 to eni 7 SNUs; the eni 1 plot is missing as the chart can only represents 6 graphs in each diagram. As can be observed, all the units demonstrate a minimum voltage exceeding 12.3 V, which denotes a satisfactory charging state. In this period there is also a slight decrease in the minimum battery voltage value, showing an energy imbalance between the primary and secondary sources, mostly due to sunlight reduction, as in the graph for the ENUs.

Detailed information about charge status and trending are also available; Figure 8 gives an example of how the current drawn by or supplied to the battery is compared with charge status. As shown here, the energy balance keeps the battery voltage at a steady, satisfactory level. Extensive data logs and reports are available to help the maintenance team

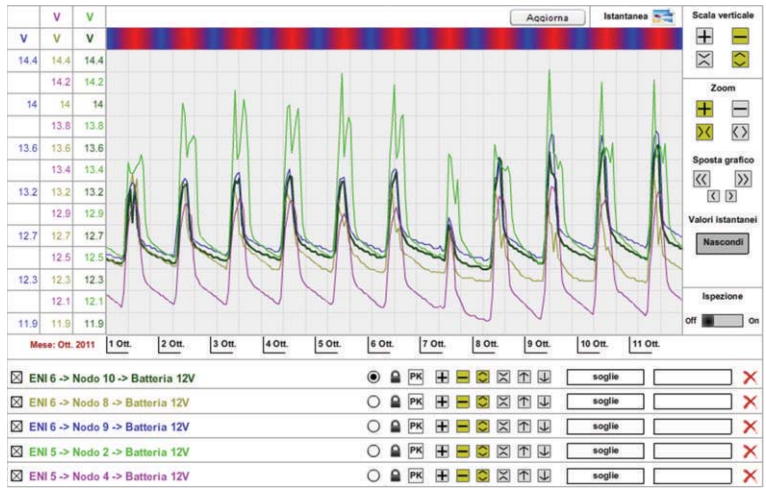


Figure 6: Battery voltage of the ENUs of the eni 6 and eni 5 subnetworks from October 1st to October 11th.

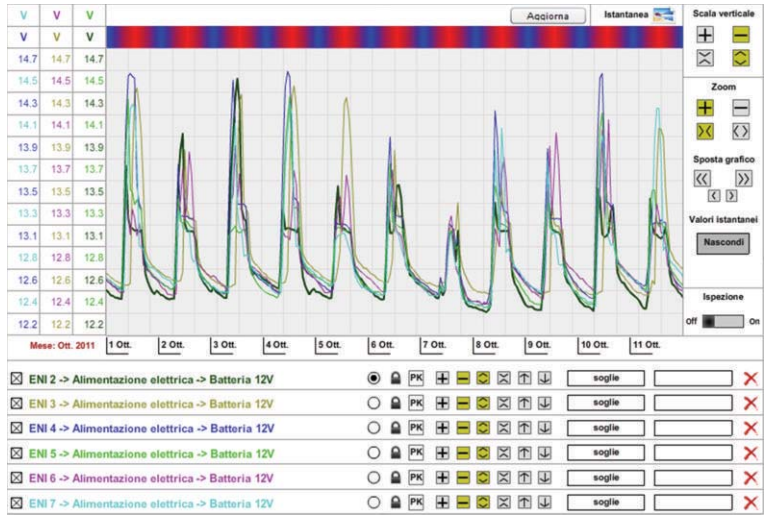


Figure 7: Battery voltage of SNUs 2 to 7 from October 1st to October 11th.

in evaluating any critical event or servicing required to keep the system in full operation.

8.3. VOC Detector Energy Budget. Continuous power-on operation requires a 35 mA h charge, which corresponds to about 2 months of full operation with a 60 A h primary energy source. However, as described above, this can be lengthened due to PVP integrated power. Furthermore, since the UV lamp's expected life is more than 6000 hours of continuous operation, that is, about four months, routine maintenance for the system—UV lamp replacement, PID refitting and battery replacement—can be planned on a four-month cycle.

9. Experimental Results

Data from the individual sensors deployed on the field can be directly accessed and presented in various formats. First, the data from the field is forwarded to a central database for storage; then, for effective data rendering the system offers a web-based interface, allowing us to process and view data in real time. There are several formats for displaying the main factors, such as weather information and VOC concentrations. However, it is also possible to access raw data, and generate summary reports relating to specific time periods and specific network areas. Furthermore, all monitored variables may be related to each other visually. Following are examples of the various formats the interface offers.

The overview map (Figure 9(a)) provides information about the entire network installed at the PEM plant. Each SNU is represented with a gray circle and a blue arrow indicating the wind direction. By selecting one of the circles (Figure 9(b)) a summary panel appears that lists the current temperature, humidity, wind speed and direction, as well as VOC concentrations over a daily, weekly, or monthly period. Minimum and maximum values of the day are also shown.

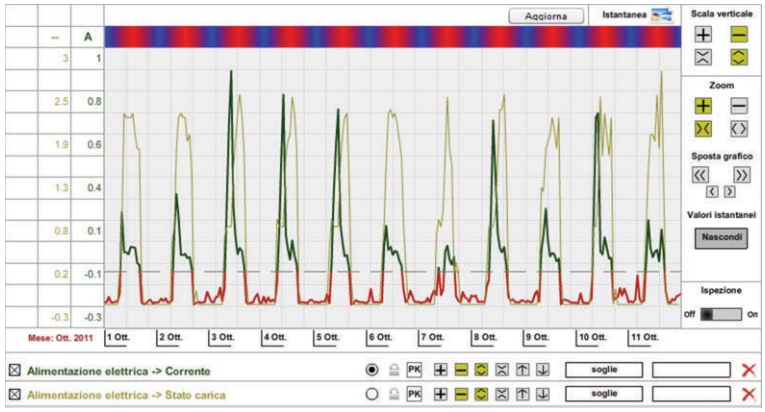
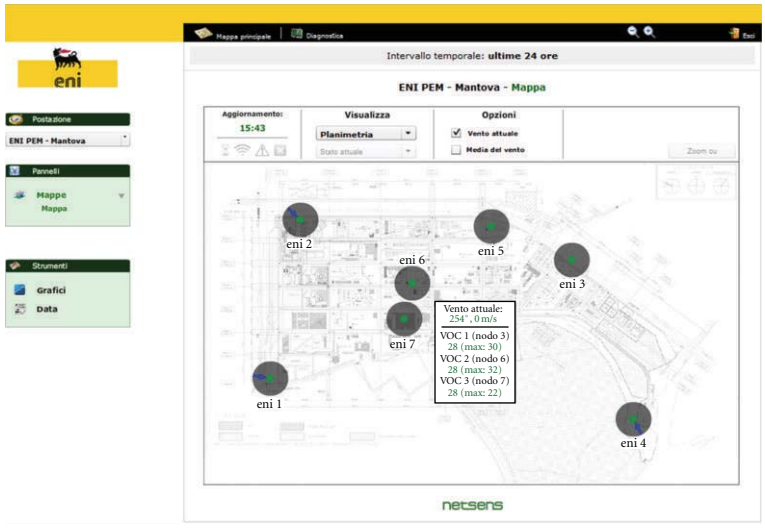
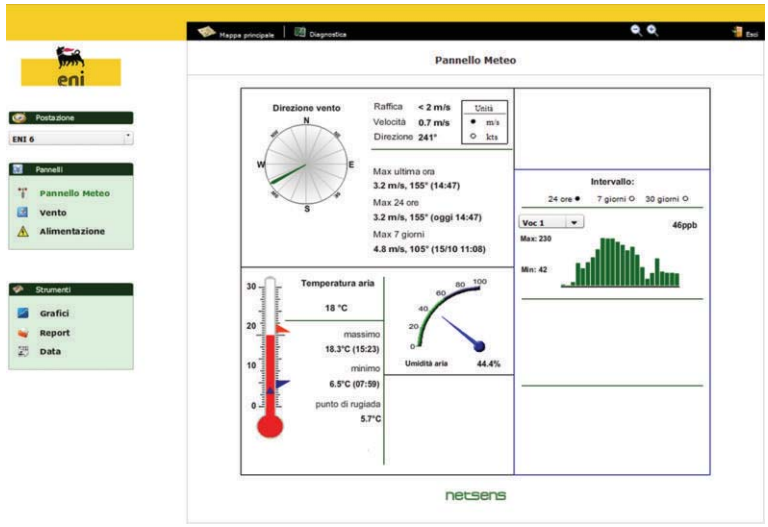


Figure 8: Current draw and charge status of SNU 1 from October 1st to October 11th.

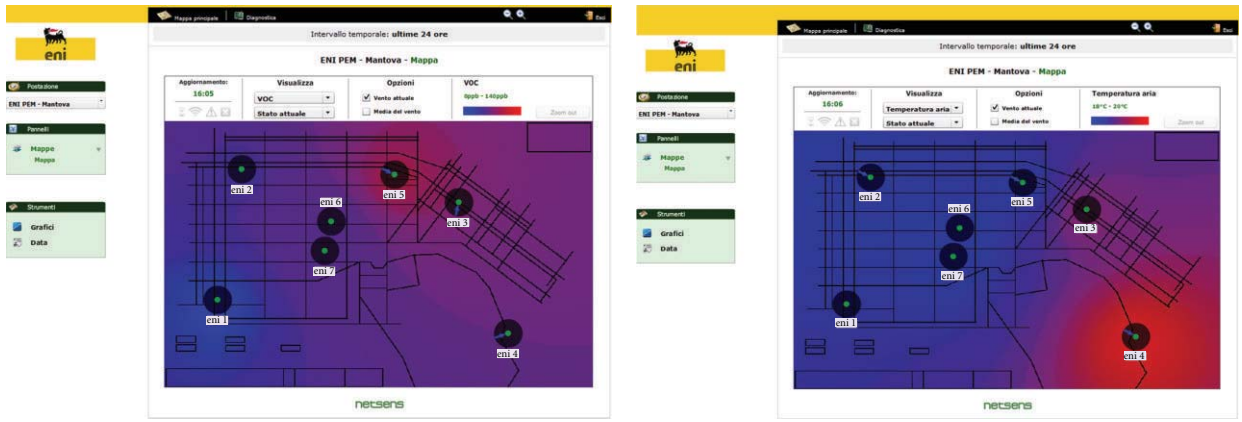


(a)



(b)

Figure 9: Examples of data rendering—installation map (a) and summary panel (b).



(a) (b)

Figure 10: Examples of data rendering—VOC concentration (a) and temperature distribution (b) across plant site.

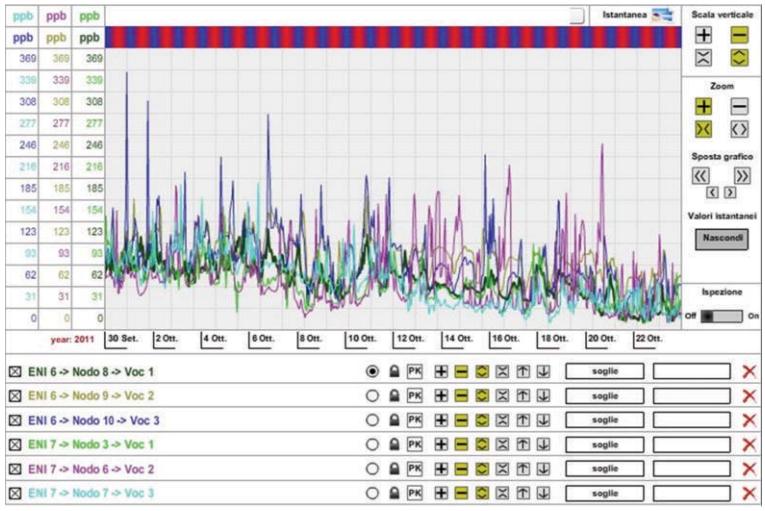


Figure 11: Example of data rendering—graph of VOC concentration in six detectors deployed around the chemical plant.

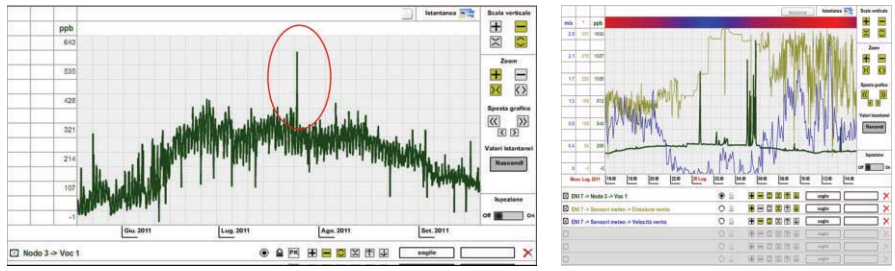
Figure 10 shows a different possible type of 2D representation of the VOC, (a), and climatic, (b), distribution over the plant.

These representations, obtained using an interpolation algorithm, are in pseudo-colour. Blue denotes a lower concentration/temperature, while red indicates a higher one. This format is particularly useful for quickly identifying quickly if there are any areas within the plant with high VOC concentrations and whether they are related to meteorological conditions. It should be emphasized that the choice of red was merely a chromatic one; it has absolutely no reference to any risky or critical condition.

An additional visual rendering of the data gathered by each sensor on the field can be obtained by opening the graphic panel window, see Figure 11. This panel allows anyone to display graphically the stored data in any time interval; up to six different and arbitrarily selected sensors can be represented in the same window, for analysis and comparison. The graph in Figure 11 shows the trend in VOC concentration values detected by the six PIDs deployed around the chemical plant over one month; the background values are similar to each other, demonstrating the effectiveness of the calibration procedure.

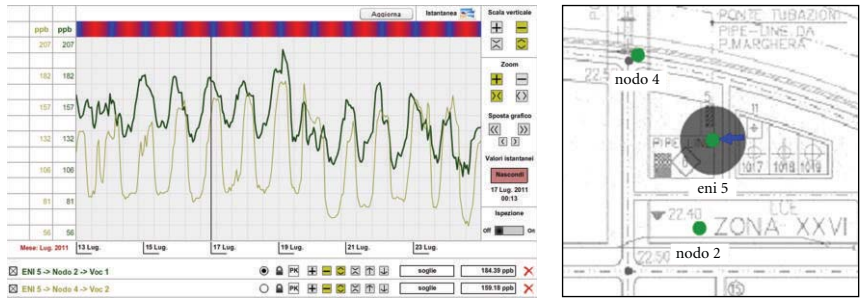
By selecting specific areas within the chart it is possible to further increase the details of the graph, allowing an easier interpretation of any short peak detected. In fact, thanks to the intensive 1-minute sample interval, the evolution of the concentration, along with other relevant weather parameters, can be accurately displayed. Moreover, a useful inspection tool allows users to quickly record min, max and mean values in any selected period.

Figure 12(a) shows VOC concentration readings over a long term (4 months) from a sensor positioned along the perimeter of the industrial area. As can be observed, the data reveals an increase in the background value during the summer, probably due to higher temperatures; in fact, the peak value (a concentration greater than 500 ppb), registered around July 25th, was due to meteorological conditions that affected the dispersion of pollutants, as seen in Figure 12(b). This graph combines the peak VOC value with a very



(a) (b)

Figure 12: Graph of VOC concentration in the long term (a) and correlation between VOC and wind speed and direction in the short term (b).



(a) (b)

Figure 13: Graph of VOC concentration in the pipeline area.

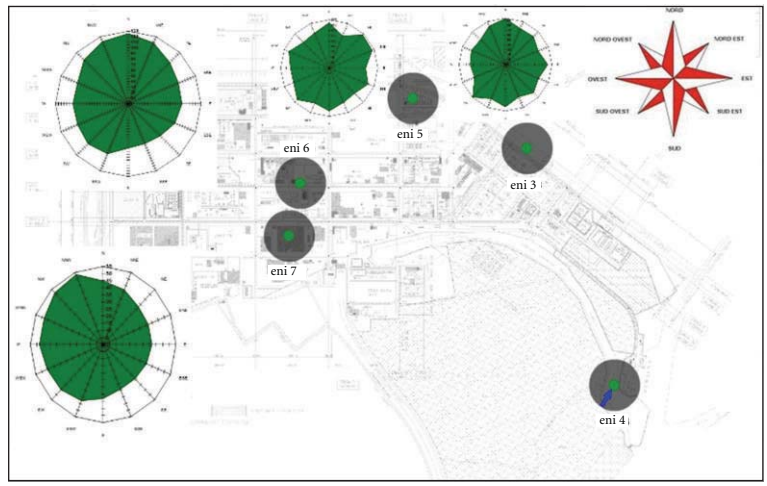
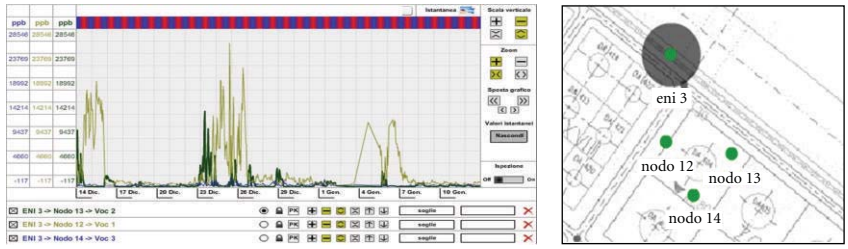


Figure 14: Example of data rendering—Correlation between wind and VOC concentration in four positions along the network.



(a) (b)

Figure 15: VOC concentration peaks detected in the benzene storage tank (floating roof).

low wind speed, conditions which favor the accumulation of pollutants from the nearby production plant. However, this was merely a local increase since the other sensors simultaneously recorded much smaller peaks. Furthermore, the values decreased from September on.

Figure 13 shows the trend of VOC concentrations in the pipeline area. Two sensors are positioned in the same area, but VOC 1 is closer to the emission sources and consequently has a higher background reading. Both sensors clearly demonstrate the cyclical effects of day and night.

When VOC sources need to be identified, the correlation between wind/speed direction and VOC concentration is vital; for this reason, a graphic representation relating these two parameters was mandatory. In Figure 14 different representations of VOC concentrations combined with the wind direction data are shown for four detectors located along the plant's perimeter; this particular plot shows the wind directions referenced to the north and the VOC concentration in ppb.

ENU 1, located in the southwest corner of the plant, is shown in Figure 14. VOC concentration is higher in quadrants I and IV, showing that the net VOC flow is entering the plant area. This may be related to the emissions generated by the traffic on the motorway running along the west side of the plant, or it may be due to emissions from other industrial sites, such as the petroleum refinery (WSW) and its storage area (WNW) located just across the motorway. In the same diagram, ENU 2, which is instead located at the northwest corner of the plant, the highest concentration values are in quadrants I and II; also in this case the VOC flow is coming from north of the plant area. In contrast, the other plots (near ENU 5) show a homogeneous distribution of VOC concentration from all directions. Hence, this format of data rendering allows PEM plant staff to quickly identify and visualize wind directions in areas of high levels of VOC concentration, while giving an overview of the predominant orientation of the VOC flux during the day.

The system's interface also allows users to keep tabs on specific areas, as in the case of the sensor network installed in the proximity of the benzene storage tank roof. In fact, data gathering was useful in identifying those phases in processing that are potentially more significant in terms of emissions. Specifically, concentration peaks are encountered with regular frequency (see Figure 15) after completely filling the tank; since, in such conditions, the floating roof is located closer to the sensors, eventual sealing problems along the perimeter of the roof due to wear or warping of the seals can be discovered. Furthermore, in the summertime the high volatility of the compounds could lead to variations in VOC concentrations during the emptying of the tank as well. Whatever the situation, the registered values, however, always are within the accepted range, since they are from a direct emissions source.

10. Conclusions

An end-to-end distributed monitoring system of integrated VOC detectors, capable of performing real-time analysis of gas concentration in hazardous sites at an unprecedented time/space scale, has been implemented and successfully tested at an industrial site near Mantova, Italy. The fundamental criteria for our system were providing the site with: a flexible and cost-effective monitoring tool that identifies emission sources in real-time year round, using easily redeployable and rationally-distributed monitoring stations that were suitable for all of the site's equipment, in order to achieve better management of abnormal situations.

Piloting the system allowed us to pinpoint additional key traits. For example, collecting data at 1-minute time intervals meets several needs: identifying short-term significant events, quantifying the emission readings as a function of weather conditions as well as of operational procedures, in addition to identifying potentially hazardous VOC sources in the plant area.

Moreover, the choice of a WSN communication platform gave excellent results, above all in allowing for redeploying and rescaling the network's configuration according to specific needs as they arise, while, at the same time, greatly reducing installation costs. Furthermore, real-time data through a web-based interface provided both adequate levels of control and quick data interpretation in order to manage specific situations. The program offers multiple formats for visualizing the data. In terms of the actual detectors, among the various alternatives available on the market, PID technology proved to meet all the major requirements, as PIDs are effective in terms of energy consumption, measuring range, cost, and maintenance once installed in the field. The energy budget was another significant element to be considered, particularly in ATEX Zone 0 areas, so secondary sources (PVPs) were adopted. Finally, fitting weather sensors at the nodes of the main network stations ensured a clearer understanding of on-field phenomena and their evolution, thus providing accurate identification of potential emission sources.

Future activity will involve, primarily, standardizing the application to allow deploying the WSN in other network industries (e.g., refineries), in addition to assessing WSN infrastructure monitoring of other environmental indicators.

Acknowledgments

This work was supported by eni SpA under contract no. 3500007596. The authors wish to thank W. O. Ho and A. Burnley, at Alphasense Ltd., for their many helpful comments and clarifications concerning PID operation, as well as E. Benvenuti, at Netsens s.r.l., for his valuable technical support. Assistance and support from the management and technical staff of Polimeri Europa Mantova is gratefully acknowledged. Furthermore, J. A. Thonn's help in revising the English was invaluable.

References

[1] G. Manes, R. Fusco, L. Gelpi, A. Manes, D. Di Palma, and G. Collodi, *Real-Time Monitoring of Volatile Organic Compounds in Hazardous Sites*, chapter 14, Intech Book, Environmental Monitoring, 2011.

[2] W. Tsujita, H. Ishida, and T. Moriizumi, "Dynamic gas sensor network for air pollution monitoring and its auto-calibration," in *Proceedings of the IEEE Sensors*, pp. 56–59, October 2004.
[3] F. Tsow, E. Forzani, A. Rai et al., "A wearable and wireless sensor system for real-time monitoring of toxic environmental volatile organic compounds," *IEEE on Sensors Journal*, vol. 9, no. 12, pp. 1734–1740, 20092009.
[4] S. Choi, N. Kim, H. Cha, and R. Ha, "Micro sensor node for air pollutant monitoring: hardware and software issues," *Sensors*, vol. 9, no. 10, pp. 7970–7987, 2009.
[5] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler, "An analysis of a large scale habitat monitoring application," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, pp. 214–226, Baltimore, Md, USA, November 2004.
[6] R. Adler, P. Buonadonna, J. Chabra et al., "Design and deployment of industrial sensor networks: experiences from the North Sea and a semiconductor plant," in *Proceedings of the ACM SenSys*, San Diego, Calif, USA, November 2005.
[7] MiniPID User Manual V1.8., IonScience, 2000.
[8] G. F. Manes, unpublished results.
[9] J. Jeong, D. Culler, and J. H. Oh, "Empirical analysis of transmission power control algorithms for wireless sensor networks," in *Proceedings of the 4th International Conference on Networked Sensing Systems (INSS '07)*, pp. 27–34, IEEE Press, Piscataway, NJ, USA, June 2007.
[10] G. Manes, R. Fantacci, F. Chiti et al., "Energy efficient MAC protocols for Wireless Sensor networks endowed with Directive antennas: a cross-layer solution," in *Proceedings of IEEE Radio and Wireless Symposium (RWS '08)*, pp. 239–242, Orlando, Fla, USA, January 2008.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 421259, 8 pages
doi:10.1155/2012/421259

Research Article

For the Pet Care Appliance of Location Aware Infrastructure on Cyber Physical System

Chung-Ming Own

Department of Computer and Communication Engineering, St. John's University, Taipei 62107, Taiwan

Correspondence should be addressed to Chung-Ming Own, cmown@mail.sju.edu.tw

Received 24 February 2012; Accepted 12 April 2012

Academic Editor: Chih-Yung Chang

"Technology stems from humanity." This sentence is a good description of modern life. The interaction between humans and physical devices and objects in the real world is gaining more attention and requires a natural and intuitive methodology to employ. Cyber physical systems refer to a new generation of systems that integrate computational and physical capabilities and are capable of interacting with humans through many new modalities. Thus, this study examines the ability of computation, communication, and control technologies to improve human interaction with pets. This study proposes a mobility-aware algorithm to enable digital home technology for pets and implements the proposed system to prove that it meets the needs of pet owners.

1. Introduction

Over the years, system and control researchers have pioneered the development of powerful system science and engineering methods and tools, such as time and frequency-domain methods, state space analysis, system identification, filtering, prediction, optimization, and stochastic control. Simultaneously, scientific researchers have made major breakthroughs in many innovative approaches to ensure computer system reliability, security, and efficiency. Cyber physical systems (CPSs) have attracted the attention of many researchers. These systems aim to develop new science and supporting technology by integrating knowledge and engineering principles across computational and engineering disciplines [1, 2].

Unlike traditional systems [3], CPS is highly heterogeneous because it interconnects many heterogeneous cyber and physical devices into an application module, including computers, various types of sensors, and actuators. Thus, the technology for communicating between the physical world and computers is becoming increasingly important.

Wireless sensor networks (WSNs) consist of a large number of unattended, self-organized microsensors scattered in an area for a specific application. Each microsensor can sense environmental data, perform simple computations, and transmit data over a wireless medium to a command center, either directly or through a cluster gateway. Although WSNs are similar to networks, they differ from traditional networks primarily because of their strict energy constraints, greater sensor node density, lower cost, and precision design for information gathering [4].

Recent localization research involving location-aware applications has concentrated on improving the accuracy of target locating [5]. Energy efficiency and target tracking are essential for WSN deployment. The contradicting goals of energy saving and communicating activities make it difficult to determine the rate of triggering the localization system. This study shows that the energy consumption of a mobile unit is related to the sampling rate of the location information.

Because of the low birth rate in Taiwan, researchers should focus more on the rapid changes in lifestyle. Some studies indicate that when people go to a park on the weekend, more of them are walking dogs than carrying babies. According to a June 2006 report from Pet Care Services in the United States, America's pet care service generated approximately 363 million dollars in revenue in 2005. The annual growth rate of this service should reach 6% in the future [6]. In addition, Eastern Europe and Asia will become potential markets for pet services [7]. Meadows and Flint indicated that a low birth rate and weakening links between family members have increased the importance of pets [8], leading to a corresponding rise in pet services.

This study attempts to improve pet appliances with the ability of location-awareness and to help pet owners raise their pets easily. The proposed system integrates the concepts of CPS and WSN communication. The mobility of pets can be influenced by the location. That is, the mobile units in a WSN are classified as live, normal, or slow depending on their location. The parameters of velocity, moving distance, and stationary time vary greatly. This study presents a fuzzy logic system based on this concept. The contribution of this study is twofold. First, this study proposes a location-based mobility-aware sampling mechanism based on status classification. Second, this study applies a fuzzy logic system for decision making. Simulation results show that the proposed scheme can extend network lifetime.

This paper is organized as follows: Section 2 provides a brief review of pet products and WSN modules; Section 3 presents the proposed system model; Section 4 provides a discussion on the implementations; and, finally, Section 5 offers a conclusion.

2. Preliminaries

2.1. Pet Raising Solution. As Taiwanese society continues to have a low birth rate and an aging population, increasingly more people regard their pets as family members. This trend is reflected in pet-related products and market activities. For example, some pet owners have started bringing their pets with them while traveling. A report from the 2010 Asia Pacific Pet Economic Conference [6] mentioned that the pet industry has grown considerably in recent years. They forecasted that the market would double in the following two years. According to previous research [7], families in Taiwan raised 1,630,000 dogs in 1999. However, this figure decreased to 1,320,000 in 2007. The family average has 1.55% dogs. Conversely, only 195,000 cats were in families in 2001, and the total increased to 281,000 in 2006. This is an increase of 4.4% per family. Based on these figures, the average family has 1.6% cats. According to a report of the council for economic planning and development, more than 166,000 babies were born in Taiwan in 2009, representing a drop of more than 20,000 from the previous year. Thus, Taiwan's birthrate has dropped to the world's lowest at 8.29%, with only under one baby born per woman over a lifetime. The average family has more pets than children. This means that the demand for pet products will grow quickly, and household spending on pets will exceed that for children. The pet industry and pet owners have gradually begun to realize the need for automated raising devices.

Pet doors are one of the most common products in the market, and various types of pet doors are available for cats and dogs. Pet doors can be fitted in a lower portion of a wall or an existing full-sized door. A pet door may consist simply of a flap hung from a horizontal axis. This flap swings open, against the force of gravity, when pushed by an animal. A simple latch may hold the door in a closed position to prevent the movement of the door in either direction. The problem with this simple construction is that any animal small enough to fit through the opening may gain entry or egress, depending on the position of the latch. To prevent passage of unwanted stray animals, electronic pet doors have been designed with magnetically operable latches. In this type of design, any magnetic tag of adequate field strength can unlock the latch.

Pets face many of the same problems as humans, such as obesity, diabetes, and stomach problems. Automated feeding machines can provide for the care of such pets. Several automated pet feeders on the market are capable of dispensing kibbles, and some are capable of feeding canned food. Feeding kibbles do not cause food spoilage, whereas feeding canned food does. Canned food cannot be left in a device for prolonged periods because it spoils.

Pets also tend to be restless when owners are not at home, and they become hyperactive when owners return. Thus, a feeding device can help reinforce the behavior of playing when owners are not at home. It is occasionally necessary for pet owners to reinforce pet behavior by providing food for certain behaviors they may want the pet to perform.

There are many automatic pet feeders for feeding pets at predetermined times during an owner's absence. This type of pet feeder comprises a base, a feeding bowl with pie-shaped divisions, a timer module, a bowl cover, handle to bowl cover, and a locking mechanism to hold the entire unit in place. The timer provides programmed feeding schedules that determine the time the bowl cover closes or opens. This programming can be achieved through the timer interface or RF and IR remotes.

2.2. WSN Module. A typical WSN consists of sensors containing nodes with varying capabilities that collaborate with each other. Current WSNs include several advanced research modules with variable sensor sizes, power consumption, operating systems, and basic sensing abilities. The Mica family of sensors is one of the most common sensing modules in use [9, 10]. These sensors are supported by numerous operating system and sensing modules, including TinyOS, Mantis OS, and Contiki. The Mica family includes the MicaZ, Mica2, and Mica2Dot series of sensors.

The Telos family of sensors consists of TelosA and TelosB motes [11]. The Telos sensors represent a newer generation of motes when compared to the Mica family, because they have a USB interface for data collection and programming. The Tmote sky sensors contain a USB port to facilitate programming and are an exact replica of the TelosB suit of sensors. These features make them well-suited to experiments involving wireless sensor networks.

The Crossbow Imote2 is a sensor module in the Intel PXA271 Xscale processor with a built-in 2.4 GHz antenna [12]. It is a powerful module that supports computationally intensive tasks such as digital image processing. This is because of the scaling capabilities of its processor, which range from 13 up to 416 MHz. The 256 KB of on-chip SRAM, 32 MB of SDRAM, and 32 MB of FLASH memory in this module provide several orders of magnitude more resources for memory-intensive applications than other modules. Power consumption is extremely low, making this

module ideally suited for demanding but battery-powered applications.

SHIMMER is a sensor module for health-related technologies with intelligence, modularity, mobility, and experimental reusability [13]. This module supports wearable applications such as capturing real-time kinematic motion and physiological sensing. SHIMMER motes are driven by TinyOS and support up to 2 GB of data storage for offline data capture. Some applications of this module include sleep studies, cognitive awareness, vital signs monitoring, and chronic disease management.

The proposed system is based on the Octopus family [14]. Octopus-I includes an 802.15.4 compliant RF transceiver and works around the ISM band (from 2.4 to 2.48 GHz) with a direct sequence spread spectrum. Octopus-I is also programmed using TinyOS, and the sensor node design is based on the Atmel ATmega128L. The programming board provides a serial interface and an onboard microcontroller for extended monitor control. The other module is Octopus-II, which is a reliable low-power WSN module for extremely low power, high data rate, sensor network applications. This module has 10 Kb of on-chip RAM, is compatible with the IEEE 802.15.4 protocol, and has an integrated on-board antenna. Octopus-X is a new WSN module in this family, the MCU is CCC2431, and the external crystal is up to 32 MHz.

2.3. Fuzzy Logic System. Fuzzy logic systems (FLSs) are knowledge-based or rule-based systems. The heart of a fuzzy system is a knowledge base consisting of the so-called fuzzy IF-THEN rules. A fuzzy IF-THEN rule is an IF-THEN statement in which some words are characterized by membership functions. The basic configuration of a pure fuzzy system is shown in Figure 1. The fuzzy rule base represents the collection of fuzzy IF-THEN rules. When an input is applied to a system, data from a real-value point is mapping to a fuzzy set in [0, 1], and the inference engine computes the output set corresponding to each fuzzy rule. The defuzzifier then computes a crisp output from these rule output sets.

Consider a p-inputs 1-output FLS, using singleton fuzzification, center-of-sets defuzzification and "IF-THEN" rules of the form

$$R_l : \text{IF } x_1 \text{ is } F_1^l \text{ and } x_2 \text{ is } F_2^l \text{ and} \ldots \text{and } x_p \text{ is } F_p^l, \quad \text{THEN } y \text{ is } G^l. \tag{1}$$

Assuming single fuzzification, when an input $x' = \{x_1', \ldots, x_p'\}$ is implemented, the degree of firing corresponding to lth rule is defined as

$$\mu_{F_1^l}(x_1') * \mu_{F_2^l}(x_2') * \cdots * \mu_{F_p^l}(x_p') = \prod_{i=1}^{p} \mu_{F_i^l}(x_i'), \tag{2}$$

where $*$ and Π both indicate the t-norm [15], $\mu_{F_i^l}(\cdot)$ is represented as the membership function of the ith item in the l fuzzy rules. Accordingly, the center-of-sets defuzzifier is used to generate the output of FLS by computing the centroid, c_{G^l}, of every consequent set G^l, and then computing a weighted average of these centroids. The weight corresponding to the

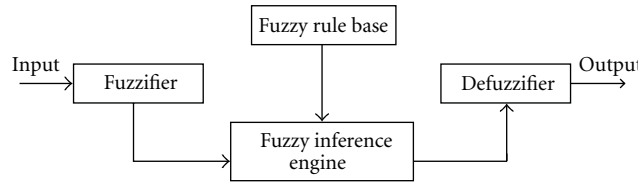


Figure 1: The structure of a fuzzy logic system.

lth rule consequent centroid is the degree of firing of the lth rule, $\Pi_{i=1}^{p} \mu_{F_i^l}(x_i')$, that is, the general inference system is defined as

$$y_{\cos}(x') = \frac{\sum_{l=1}^{M} c_{G^l} \Pi_{i=1}^{p} \mu_{F_i^l}(x_i')}{\sum_{l=1}^{M} \Pi_{i=1}^{p} \mu_{F_i^l}(x_i')}, \tag{3}$$

where M is the number of rules in the FLS. $y_{\cos}(\cdot)$ is used to derive the output of the center-of-sets defuzzifier, which value is indicated as the importance of inference results [15]. This equation is represented as our fuzzy inference engine in this study.

Accordingly, in our real world, knowledge about exceptions may be incomplete, so representing the set of exceptions exclusively by using a default rule is impossible. Therefore, reasoning systems should be modified to accommodate such partially true knowledge. In our study, the authors use fuzzy rules of the following type for the rule weight specification

$$R_l' : \text{IF } x_1 \text{ is } F_1^l \text{ and } x_2 \text{ is } F_2^l \text{ and } \ldots \text{and } x_p \text{ is } F_p^l, \quad \text{THEN } y \text{ is } G^l \text{ with } CF_l, \tag{4}$$

where CF_l is a certainty grade (i.e., rule weight). Thus, the weighted inference equation is derived by the previous equation (3):

$$y_{\cos}(x') = \frac{\sum_{l=1}^{M} c_{G^l} \Pi_{i=1}^{p} \mu_{F_i^l}(x_i') \cdot CF_l}{\sum_{l=1}^{M} \Pi_{i=1}^{p} \mu_{F_i^l}(x_i')}. \tag{5}$$

3. System Design

Although numerous automatic pet monitoring systems exist, such as automatic pet doors and pet feeders, these systems cannot meet the needs of pet owners. For example, most pet care systems are based on infrared detector/recognition, which can be used to spot pets at the door, register their movements, and alert owners when pets enter areas where they are not allowed. This type of design has some disadvantages because infrared detectors can be influenced easily by unknown reasons. The detection cannot go through. The proposed system focuses on the implementation of a pet care system with a location-aware algorithm.

3.1. Intelligent Pet Door. Many families install a pet door for family pets. Pet doors are adapted to be fitted in the lower portion of a wall or existing full-sized door. Pet doors allow family pets to pass the door easily. Figure 2 shows the system

flowchart of the pet door. The pet wears a sensing tag on its collar, and the collar broadcasts the pet ID continuously. Figure 3(a) shows the collar design. The pet door detects the pet's location and periodically requests the pet schedule from the server. The proposed system is based on the following ideas.

(1) An animal detector can detect an animal seeking passage through the pet door.

(2) A controller enables selective passage.

(3) A selective latch is disabled to allow passage past the pet door.

(4) A clock allows the pet owner to manage the time schedule or set up predefined pass-through condition for pets.

(5) A detecting tag is programmed by the WSN module, not infrared detection or magnetic detection.

(6) An LED light can inform the pet owner where the pet is located.

The development process is listed as follows. First, we installed several WSN modules as the outdoor sensing platform. These modules provide the pet owner with outdoor temperature/humidity data, allowing them to can change the pet scheduling permission according to the weather change. For example, no pets are allowed to go outside on a rainy day.

Second, we installed a light motion sensor on the top panel of the pet door to detect flap switching. By identifying the flap switching direction, the system can determine whether the pet is trying to enter or exit, illuminating differently colored lights to indicate the pet's location. The pet owner can quickly identify the pet location by reading the indicator.

Finally, according to the system diagram shown in Figure 2, a pet wears a sensing tag on its collar and broadcasts the pet id message to the pet door for every 20 seconds. This tag functions as a tracking node in Figure 3. In the meantime, the environment center also broadcasts the microtemperature message of the outdoor for 30 minutes. Finally, the control center functions as a data center collecting all the system messages from the above devices. Besides, Figure 3 is the prototype of pet tag we will load the system with the box for the function of waterproof in the actual operation.

3.2. Intelligent Pet Feeder. Most pets eat dry food. Pet owners cannot leave out wet food because it spoils in the time the pet must eat it. Dry food feeders can keep dry food fresh longer and allow the owner to feed a pet automatically, even when outside the home. These feeders are nearly always placed on the floor next to an external wall where they are accessible to crawling insects, such as ants, which have a propensity to seek a supply of food. These insects not only contaminate the pet food, but are often intolerable to the pet owner.

Numerous attempts have been made to design a pet feeder that resolves the problem of crawling insects. Most of these efforts have produced feeders with a multiplicity of discrete components. These feeders must be disassembled for cleaning and then reassembled before further use. The

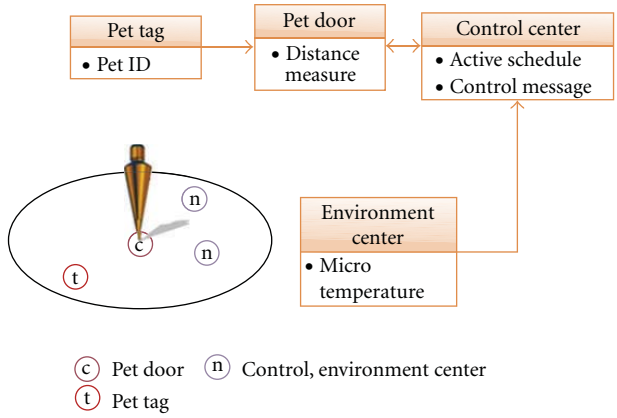


Figure 2: The system diagram of pet door.

(a)

(b)

Figure 3: (a) Sensing tag on the cat neck, (b) sensing tag on the collar.

pet feeder of a trading company has the general appearance of a conventional feeder, but has a moat-forming cavity surrounding the food bowl [16]. This pet feeder can keep the pet food and water clean until the pet is ready to eat. This pet feeder also has a bowl cover that opens and closes automatically. The bowl cover is actuated by an infrared proximity sensor and battery-operated electric motor. The sensor detects the presence of the pet and then opens the cover, enabling only the pet to have access to the food. When the pet is out of sensor range, the bowl cover closes automatically. This keeps dust, flies, and bugs from reaching the food and keeps the food fresh.

The intelligent pet feeder system in this study is based on the following ideas.

(1) An animal detector can detect an animal trying to access the food.

(2) A controller allows a permitted pet to access the food.

(3) A clock allows a pet owner to create a time schedule or establish predefined eating conditions for pets.

(4) A detecting tag is programmed by WSN module, and not infrared detection or magnetic detection.

(5) A pet owner can schedule eating time remotely.

The system process is listed as follows. First, a pet owner can use a web page to remotely modify the pet eating schedule, including the number of meals, beginning/ending time of meals, and which bowl to open immediately. Hence, when it is meal time, the feeder plays alert music to call the pet within the WSN sensing range to obtain food. A controller requests permission from the server node to open the cover. During meal time, the feeder opens the cover whenever the pet is within sensing range.

Second, if the owner forgets to feed the pet, the owner can use remote access to open the bowl on demand.

Finally, Figure 4 shows the system diagram of the pet feeder. A pet wears the sensing tag on the collar as the description in Figure 2, and this tag broadcasts the pet ID every 20 seconds to identify the pet's location. The control center functions as a data center, collects all the system messages from the above devices, and includes the eating schedule and the control messages.

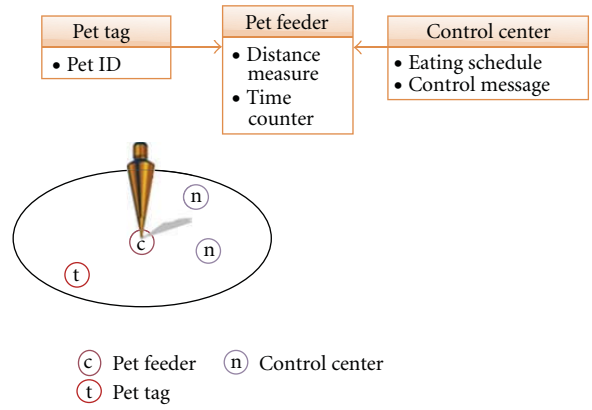


Figure 4: The system diagram of pet feeder.

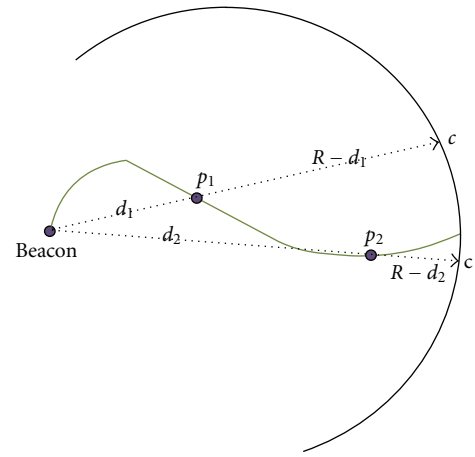


Figure 5: The node illustration in WSN environment.

3.3. Location Aware Algorithm. This algorithm helps the system detect the exact time the pet reaches the critical point. Because of the high speed of home pets, the system must detect the pet's location rapidly to react to the pet's activity.

The critical point is where the line of movement intersects with the critical region. To simplify the sampling mechanism, Figure 5 shows a WSN environment for a one-dimensional space. Figure 5 shows two recent sample points p_1 and p_2. As the mobile unit reaches position p_2, the proposed system computes the next sample time by calculating the time for the mobile unit to move from the current position p_2 to the critical point c in velocity V_{p_2}, which is estimated by dividing the rate of movement from p_1 to p_2. The critical region is set as the distance away more from than R distance to the beacon node in WSN, and the critical points all fall on a circle, the center of which is the beacon node. Assuming that the mechanism of deriving the distance between the mobile unit and beacon node is clear and precise, the proposed system focuses on reducing the amount of distance estimation messages. During the message exchange stage, the beacon node sends probing messages to the mobile unit and waits for reply messages from the mobile unit. Both nodes can derive the relative distance by using the distance estimation method (e.g., d_1 and d_2 in Figure 5).

Accordingly, after obtaining the outgoing velocity, distance between the pet and the critical point, stationary time, and stationed location, we listed the relationships with the sampling rate (frequency) in Table 1. After calculating the velocity of a mobile unit, the sampling rate (probing frequency) increases in conjunction with the velocity because the system regards the target as active and ready to leave. When the pet approaches the critical point, the critical distance decreases and the sampling rate increases to catch the target motivation (e.g., $|R - d_1|$ in Figure 5), and vice versa. Similarly, with longer stationary time of the mobile unit and the role matching degree of the stationed location, the higher value means that the sampling rate is set low for lower activities.

The proposed inference algorithm sets up fuzzy rules for tracking the target node based on the following descriptors:

(1) distance between the mobile unit and critical point,

(2) its stationary time,

(3) its relative velocity.

This study uses three linguistic variables to represent the distance between a target node and the destination: *near*, *moderate*, and *far*. The terms representing its stationary time and relative velocity are *short*, *moderate*, and *long*, and *slow*, *moderate*, and *fast*. The consequent (i.e., the estimated time before the node reaches the critical point) is divided into

Table 1: Sampling rate with the parameters of system.

	Relative velocity	Critical distance	Stationary time	Stationed location
Sampling rate↑	Fast	Near	Short	Unfitted
Sampling rate↓	Slow	Far	Long	Fitted

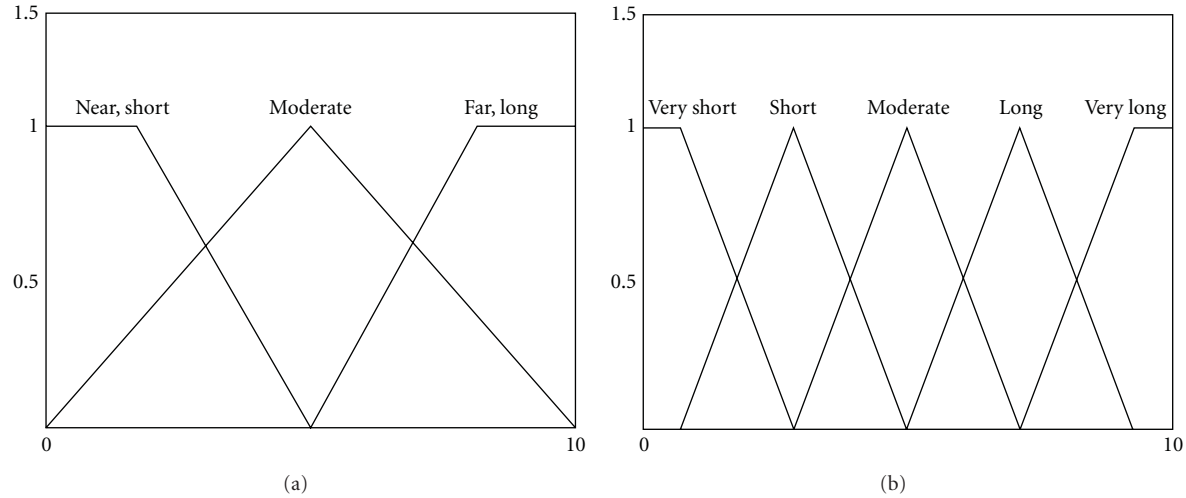


Figure 6: The MFs used to represent the linguistic labels. (a) MFs for antecedents, and (b) MFs for consequent.

Figure 7: The control interface on the web server (C# code).

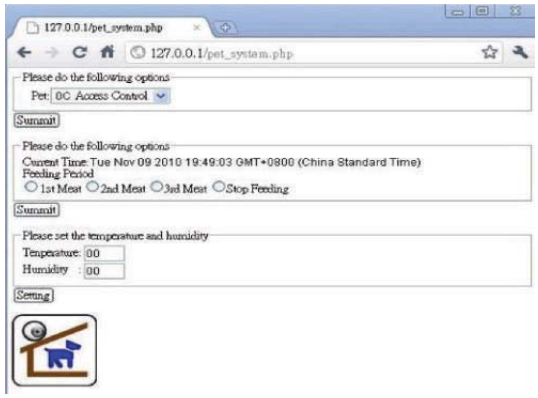


Figure 8: The control interface of the web server (php code).

five levels: *very long*, *long*, *moderate*, *short*, and *very short*. Accordingly, the proposed FLS has 27 rules.

Conversely, the value of stationed location is defined as the rule weight of the FLS. This value varies with the WSN environment. Therefore, this study uses trapezoidal membership functions (MFs) to represent near, short, slow, far, long, and fast. Triangular MFs represent moderate in Figure 6(a), and Figure 6(b) shows the MFs for consequent MFs. For every input (x_1, x_2, x_3), the output can be computed as (5):

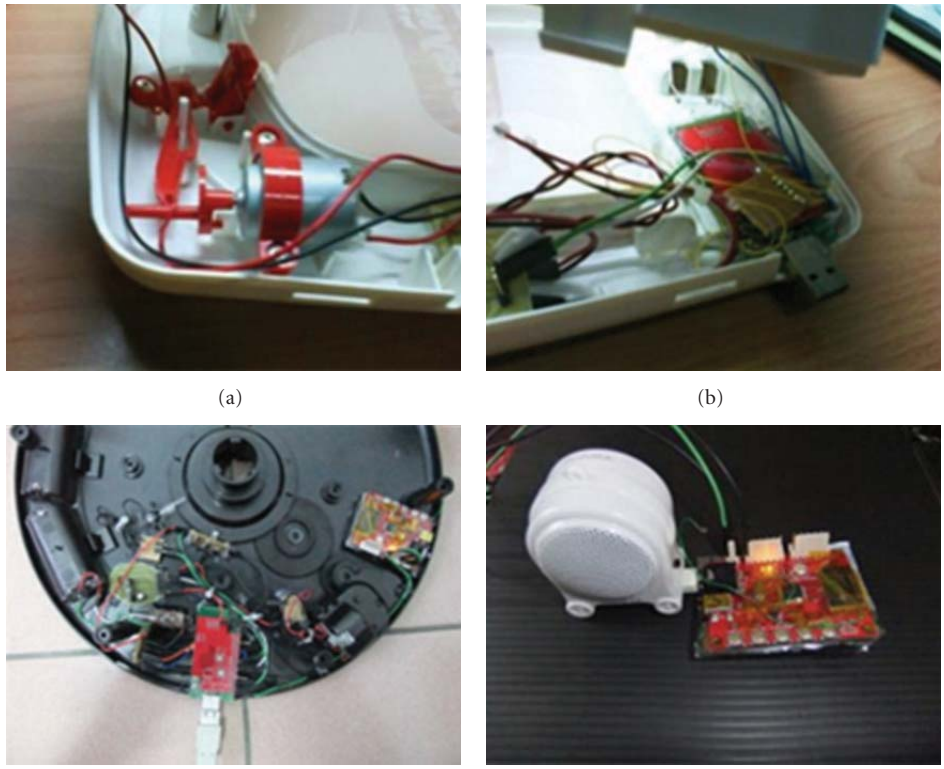
$$y' = \frac{\sum_{l=1}^{27} c_{G^l} \mu_{F_i^l}(x_1) \mu_{F_i^l}(x_2) \mu_{F_i^l}(x_3) \cdot CF_l}{\sum_{l=1}^{27} \mu_{F_i^l}(x_1) \mu_{F_i^l}(x_2) \mu_{F_i^l}(x_3)}, \quad (6)$$

where c_{G^l} is the centroid of the consequent set, and CF_l is the rule weight. The system output y' represents the possibility that the mobile unit moves across the critical point. Thus, the expect time elapsed for the mobile unit is defined as

$$T = \frac{|(R - d_b)|}{|V| \cdot y'}, \quad (7)$$

where d_b is the distance between the beacon node and mobile unit, and the absolute value on $R - d_b$ is computed as the critical distance. The variable y' is the moving possibility of a mobile unit, and the expected time increases as this possibility decreases.

After identifying the relative parameters and deriving the system output, the system obtains the time interval T to send the next probing message, and then returns this message to the beacon node, along with the reply message. The beacon node then schedules the next probing message. Both sides can then immediately suspend radio communication to conserve energy.



(a) (b)

(c) (d)

Figure 9: (a) and (b) are the disassemble pictures of pet door, (c) and (d) are the disassemble pictures of pet feeder.

4. The System Implementation

Figures 7 and 8 show the proposed control software in the control/web server. The proposed design uses a C# program for the native client user (Figure 7), and a PHP program can be remotely executed on a smart phone as the pet owner wishes (Figure 8). According to the software design, the pet owner can monitor environmental data (i.e., obtain the weather report), the managed pet ID, and the pet location (inside/outside). The pet owner can also set the eating time schedule and command the bowl cover to open remotely.

Figure 9 shows a disassembled pet door. Figure 9(a) shows that we directly connected the motor to the IO interface of the WSN module, and Figure 9(b) shows that we installed the Octopus-II as the coordinator node into the system design. This node also codes the location-aware algorithm. Figure 9(c) shows the interior of the pet feeder. The Octopus-II node controls the gears to rotate the bowl and open or close the cover. Figure 9(d) shows the music module. At each set eating time, the server sends data packets to the WSN module, causing the music module to play music when it is eating time.

5. Conclusion

The interaction between humans and physical devices and objects is attracting increasing attention. Many studies have attempted to provide a natural and intuitive approach to request services. The current trend of combining pet control and CPS technology offers exciting future developments. This study presents an intelligent pet care system based on the concept of the Internet of things. The proposed system is based on smart-home technology, including an intelligent pet door and pet feeder. The implementation reported here shows that the system can overcome the disadvantage of traditional products and meet the needs of pet owners.

Acknowledgment

The author thank the National Science Council of the Republic of China, Taiwan, for partially supporting this research under Contract no. NSC 100-2218-E-129-001.

References

[1] E. Lee, "Cyber-physical systems—arecomputing foundations adequate?" *Workshop On Cyber-Physical Systems*. In press.

[2] L. Sha, S. Gopalakrishnan, X. Liu, and Q. Wang, "Cyber-physical systems: a new frontier," *Machine Learning in Cyber Trust*, pp. 3–13, 2009.

[3] Y. Tan, G. Goddard, and C. P. Lance, "A Prototype architecture for cyber-physical systems," *ACM SIGBED Review*, vol. 5, no. 1, 2008.

[4] K. R. Bhakare, R. K. Krishna, and S. Bhakare, "An energy-efficient grid based clustering topology for a wireless sensor network," *International Journal of Computer Application*, vol. 14, pp. 24–29, 2012.

[5] T.-C. Chiang, J.-L. Chang, and S.-W. Lin, "A distributred multicast protocol with location-awae for mobile Ad-Hoc networks," *Advances in Intelligent and Soft Computing*, vol. 129, pp. 691–697, 2012.
[6] Packaged Facts, *Pet Care Services in the U.S*, 2nd edition, 2006.
[7] APPEC, "Asia pacific pet economic conerence," http://www.apec.org/.
[8] G. Meadows and E. Flint, *The Cat Owners'Handbook*, 2006.
[9] B. Buchli, F. Sutton, and J. Beutel, "GPS-equipped wirelesssensor network node for high-accuracy positioning applications," *Lecture Notes in Computer Science*, pp. 179–195, 2012.
[10] C. Townsend and S. Arms, "Wireless Sensor Netowrks, Priciples and Applications," Principles and Applications, Sensor Technology Handbook Newness Publication, 2005.
[11] D. Wen and Y. Ma, "The Application of wireless sensor in aquaculture watrer quality monitoring," *IFIP Advances in Information and Communication Technology*, vol. 370, pp. 502–507, 2012.
[12] Crossbow Imote2, http://www.xbow.com/.
[13] Shimmer, "Wearable Wireless Sensor Module," http://www.shimmer-research.com/.
[14] Octopus Series Family, http://www.wsnc.ntu.edu.tw/.
[15] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Direction*, Prentice Hall, New Jersey, NJ, USA, 2001.
[16] The Pet Feeder, http://gadgets.softpedia.com/gadgetsImage/Automatic-Pet-Feeder-1-22503.html.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 724846, 19 pages
doi:10.1155/2012/724846

Research Article

Multidimensional Sensor Data Analysis in Cyber-Physical System: An Atypical Cube Approach

Lu-An Tang,[1] Xiao Yu,[1] Sangkyum Kim,[1] Jiawei Han,[1] Wen-Chih Peng,[2] Yizhou Sun,[1] Alice Leung,[3] and Thomas La Porta[4]

[1] *University of Illinois at Urbana-Champaign, Urbana, JL 61801, USA*
[2] *National Chiao Tung University, Hsinchu 30010, Taiwan*
[3] *BBN Technologies, Cambridge, MA 02138, USA*
[4] *The Pennsylvania State University, Philadelphia, PA 16802, USA*

Correspondence should be addressed to Lu-An Tang, tang18@uiuc.edu

Received 16 December 2011; Accepted 21 March 2012

Academic Editor: Chih-Yung Chang

Cyber-Physical System (CPS) is an integration of distributed sensor networks with computational devices. CPS claims many promising applications, such as traffic observation, battlefield surveillance, and sensor-network-based monitoring. One important topic in CPS research is about the atypical event analysis, that is, retrieving the events from massive sensor data and analyzing them with spatial, temporal, and other multidimensional information. Many traditional methods are not feasible for such analysis since they cannot describe the complex atypical events. In this paper, we propose a novel model of *atypical cluster* to effectively represent such events and efficiently retrieve them from massive data. The *basic cluster* is designed to summarize an individual event, and the *macrocluster* is used to integrate the information from multiple events. To facilitate scalable, flexible, and online analysis, the *atypical cube* is constructed, and a guided clustering algorithm is proposed to retrieve significant clusters in an efficient manner. We conduct experiments on real sensor datasets with the size of more than 50 GB; the results show that the proposed method can provide more accurate information with only 15% to 20% time cost of the baselines.

1. Introduction

The *Cyber-Physical System* (CPS) has been a focused research theme recently due to its wide applications in the areas of traffic monitoring, battlefield surveillance, and sensor-network-based monitoring [1–6]. It is placed on the top of the priority list for federal research investment in the fiscal year report of US president's council of advisors on science and technology [7].

A CPS consists of a large number of sensors and collects huge amount of data with the information of sensor locations, time, weather, temperature, and so on. In some cases, the sensors occasionally report unusual or abnormal readings (i.e., atypical data); such data may imply fundamental changes of the monitored objects and possess high domain significance. To benefit the system's performance and user's decision making, it is important to analyze the atypical data with spatial, temporal, and other multidimensional information in an integrated manner. A motivation example is shown as follows.

Example 1. The highway traffic monitoring system is a typical CPS application. With the sensor devices installed on road networks, the monitoring system watches the traffic flow of major U.S highways in 24 hours $\times$ 7 days and acquires huge volumes of data. In this scenario, one important type of atypical events is the traffic congestion. Some frequent questions asked by the officers of transportation department are (1) where do the traffic congestions usually happen in the city?, (2) when and how do they start? (3) and on which road segment (or time period) is the congestion most serious?

In such queries, the users are not satisfied merely on a database query returned with thousands of records. They demand summarized and analytical information, integrated in the unit of atypical event. The granularity of the results should also be flexible according to the user's requirements:

some officers may be only concerned with the information in recent days, whereas others are more interested in the monthly or even yearly report. However, it is hard to support such multidimensional analysis of atypical events in CPS data, partly due to following difficulties.

(i) *Massive Data.* A typical CPS includes hundreds of sensors, and each sensor generates data records in every few minutes. The CPS database usually contains gigabytes, even terabytes of data records. The management system is required to process the huge data with high efficiency.

(ii) *Complex Event*. The atypical event is a dynamic process influencing multiple spatial regions. Those spatial regions expand or shrink as time passes by, they may even combine with others or split into smaller ones. Hence the atypical events do not have fixed spatial boundaries. They are difficult to be represented by traditional models.

(iii) *Information Integration.* In many applications, the users demand integrated information for analytical purposes. For example, a transportation officer may need a monthly summary of the congestions in the city. Then the system has to measure the similarity among daily atypical events and integrate the similar ones to provide a general picture.

(iv) *Retrieving Effectiveness.* A large-scale analytical query may contain the data from hundreds of atypical events; however, not all of them are interesting to the users. The users may only prefer a few significant results, that is, the most serious events that influence large area and last for a long time. The system should distinguish such significant events in the retrieving process and emphasize them from the majority of trivial ones.

In this study, we introduce the concept of *atypical connection* to discover the atypical events and summarize them as *atypical clusters*. The atypical cluster is a model describing multidimensional features of the atypical event. They can be efficiently integrated in a hierarchical framework to form macroclusters for large-scale analytical queries. To retrieve significant macroclusters, the system employs a guided clustering algorithm to filter out the trivial results and meanwhile guarantees the accuracy of significant clusters. The data structure of *atypical cube*, which is a forest of hierarchical clustering trees, is constructed to facilitate scalable and feasible analysis. The proposed methods are evaluated on gigabyte-scale datasets from real applications; our approaches can provide more detailed and accurate results with only 15% to 20% time cost of the baselines.

This paper substantially extends the ICDE 2012 conference version [8], in the following ways: (1) introducing the concepts of atypical cube as an integrated model for multidimensional sensor data analysis in CPS; (2) proposing the techniques to process OLAP queries based on the atypical cube, including the algorithms for both the large-scale and small-scale (i.e., drill-through) queries; (3) discussing the issues of extending atypical cube to other dimensions and introducing a case study in traffic application; (4) carrying out the time complexity analysis of proposed algorithms; (5) providing complete formal proofs for all the properties and propositions; (6) covering related work in more details and including recent ones; (7) introducing the bottom-up styled cube in more details as the background knowledge; (8) expanding the performance studies on real datasets.

The rest of the paper is organized as follows. Section 2 introduces the problem formulation and system framework; Section 3 proposes the models of atypical clusters and the algorithms to construct atypical cube; Section 4 introduces the techniques to efficiently retrieve significant clusters for OLAP queries; Section 5 evaluates the performances of proposed methods on real datasets; Section 6 discusses the extensions of proposed techniques; Section 7 makes a survey of the related work, and in Section 8 we make the conclusion.

2. Backgrounds and Preliminaries

2.1. Problem Formulation. The cyber-physical systems monitor real world by sensor networks. In most cases, a sensor reports records with normal readings. If an atypical event happens (such as a congestion is detected in traffic system), the sensor will send out atypical records. The detailed atypical criteria are different according to the application scenarios and environments (e.g., the highway types and speed limits); many state-of-the-art methods have been proposed to select the trustworthy atypical records in traffic, battlefield, and other CPS data [3, 9, 10]. Since the main theme of this study is on multidimensional analysis of atypical event, we assume that the atypical criteria are given and clean atypical records can be retrieved by CPS. In fact, some of such datasets are available to public [11].

The atypical records are represented in the format of $(s, t, f(s,t))$, where the severity measure $f(s,t)$ is a numerical value collected from sensor s in time window t. Without loss of generality, we adopt the atypical duration as the severity measure in this study, since it is commonly used in many CPS applications. For example, (s_1, 8:05 am–8:10 am, 4 mins) means that sensor s_1 has reported atypical readings for 4 minutes from 8:05 am to 8:10 am. Note that, although we focus on atypical duration in this paper, the proposed approach is also flexible to adjust to other domain-specific measures.

The atypical events are dynamic processes including many atypical records. In the traffic application, the atypical event of a congestion usually starts from a single street, which can only be detected by one or few sensors. Then the congestion swiftly expands along the street and influences nearby sensors. A serious congestion usually lasts for a few hours and covers hundreds of sensors when reaching the full size. As time passes by, it shrinks slowly, eventually reduces the coverage, and finally disappears.

By observing the phenomenon of congestion, we find that those records in an atypical event are spatially close

Table 1: Example: atypical events.

ID	Atypical Records
E_A	$\langle s_1$, 8:05 am–8:10 am, 4 min$\rangle$; $\langle s_1$, 8:10 am–8:15 am, 5 min$\rangle$; $\langle s_2$, 8:10 am–8:15 am, 5 min$\rangle$; $\langle s_3$, 8:15 am–8:20 am, 5 min$\rangle$; $\langle s_4$, 8:15 am–8:20 am, 2 min$\rangle$; ...
E_B	$\langle s_3$, 6:20 pm–6:25 pm, 2 min$\rangle$; $\langle s_4$, 6:20 pm–6:25 pm, 5 min$\rangle$; $\langle s_1$, 6:25 pm–6:30 pm, 5 min$\rangle$; $\langle s_4$, 6:25 pm–6:30 pm, 5 min$\rangle$; $\langle s_5$, 6:30 pm–6:35 pm, 5 min$\rangle$; ...
E_C	$\langle s_1$, 8:20 am–8:25 am, 1 min$\rangle$; $\langle s_1$, 8:25 am–8:30 am, 5 min$\rangle$; $\langle s_9$, 8:25 am–8:30 am, 5 min$\rangle$; $\langle s_1$, 8:30 am–8:35 am, 5 min$\rangle$; $\langle s_7$, 8:35 am–8:40 am, 3 min$\rangle$; ...

and timely relevant. Hence we introduce the following definitions.

Definition 2 (*Direct Atypical Connected*). Let $r_i\langle s_i, t_i, f(s_i, t_i)\rangle$ and $r_j\langle s_j, t_j, f(s_j, t_j)\rangle$ be two atypical records in CPS, let δ_d be the distance threshold, and let δ_t be the time interval threshold. r_i and r_j are said to be *direct atypical connected* if distance $(s_i, s_j) < \delta_d$ and $|t_i - t_j| < \delta_t$.

Definition 3 (*Atypical Connected*). Let r_1 and r_n be atypical records. If there is a chain of records $r_1, r_2, \ldots, r_n$, such that r_i and r_{i+1} are direct atypical connected, then r_1 and r_n are said to be atypical connected.

Based on the above concepts, we formally define the atypical event as follows.

Definition 4 (*Atypical Event*). Let R be the set of atypical records. *Atypical Event* E is a subset of R satisfying the following conditions: (1) for all r_i, r_j: if $r_i \in E$ and r_j is atypical connected from r_i, then $r_j \in E$; (2) for all r_i, $r_j \in E$: r_i is atypical connected to r_j.

Example 5. Table 1 shows three atypical events. Each event contains hundreds of atypical records; part of their records are listed in the second column.

Our task is to find out atypical events and integrate them in a data structure to support online analytical processing (OLAP) queries with multidimensional information. The *data cube* is a subject-oriented and integrated structure to support OLAP queries [12]. It organizes the data with multiple dimensions as a lattice of cells. In the cube, every cell corresponds to a degree of data summarization and stores the concrete measures for different queries. For example, a cell may store the atypical events as (Downtown LA, 8 am–9 am Oct. 10th: E_A). If a user wants to query the congestions in that morning of Downtown LA, the precomputed event E_A can be retrieved immediately to process the query.

Property 1. The atypical event is a holistic measure.

Proof. A measure is *holistic* if there is no constant upper bound on the storage size needed to describe a subaggregation [13].

Since the atypical events contain all the original records, although their number can be bounded, the sizes are still unbounded. Let us consider the worst case, in which there is a heavy snow and the traffic of the entire region is tied up through the whole day. In such case, even there is only one event, it includes all the atypical records of the sensor dataset. No constant bound of storage size can be found in this case. Hence the measure of atypical event is holistic. □

The atypical event is not feasible for data cubing because a holistic measure is inefficient to aggregate and compute [13]. A more succinct measure is thus required. The key challenge in atypical cube construction is indeed at designing such measure to model the atypical events and developing the corresponding aggregation operations.

Task Specification. Let R be the atypical dataset in CPS; the atypical cubing tasks are (1) finding out the atypical events from R, representing them with a succinct measure and aggregating the measure to construct a data cube; (2) effectively and efficiently processing the OLAP query $Q(W, T)$ with such cube.

2.2. System Framework. Figure 1 shows the overview of our system framework. The system consists of two components: the atypical cube construction module and the OLAP query processing module.

Atypical Cube Construction. This component offline builds up the atypical cube from the sensor data in CPS. The system first retrieves the atypical events from the dataset and then constructs the atypical microcluster to store the features of each individual event. The similarity of micro-clusters is measured based on the retrieved features. The system merges similar micro-clusters as macroclusters to integrate multiple events. The clusters are formed in hierarchical trees to construct the atypical cube, which will be used to help process the OLAP queries.

OLAP Query Processing. This component online processes the OLAP query. The key issue is to efficiently retrieve significant clusters in the query range. The query processing algorithm first determines the possible regions where the significant clusters might be (i.e., red-zones) and then prunes the micro-clusters locating outside those regions. Only the qualified micro-clusters are selected to generate the macroclusters as query results. For the OLAP query in small range, the system utilizes a two-stage query processing technique: first returns an approximate answer in short time and then computes the detailed query results.

We will introduce the cube construction methods in Section 3 and query processing techniques in Section 4. Table 2 lists the notations used throughout this paper.

3. Atypical Cube Construction

3.1. Bottom-up Styled Cube. Traditional methods construct the atypical cube by aggregating severity measures in a

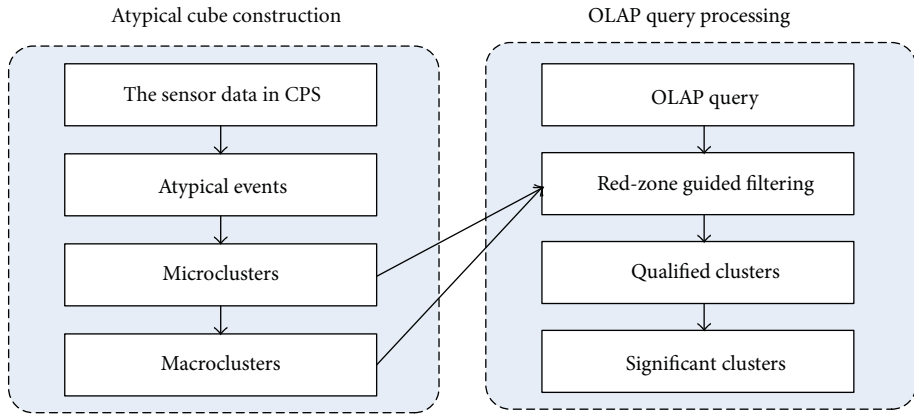


Figure 1: The overview of system framework.

Table 2: List of notations.

Notation	Explanation	Notation	Explanation
R	The CPS dataset	r_i, r_j	The atypical data records
S	The sensor set	s_1, s_2	The sensors
T	The time period	t_1, t_2	The time windows
SF	The spatial feature	TF	The temporal feature
$f(s,t)$	The severity measure	$F(S,T)$	The total severity
E_A, E_B	The atypical events	W	The spatial region
$Q(W,T)$	The analytical query	C_A, C_B	The atypical clusters
μ_i	The agg. severity by s_i	ν_i	The agg. severity by t_j
δ_t	The time threshold	δ_d	The distance threshold
δ_s	The severity threshold	δ_{sim}	The similarity threshold

bottom-up style. The hierarchies are predefined on temporal, spatial, and other related dimensions, and the severities are then aggregated following such hierarchies. For example, the severity is aggregated by hour, day, month, and year in temporal dimension. In the spatial dimension, the hierarchies are built by partitioning the data with fixed regions, such as zipcode areas, street names [14], highway numbers [15], or the R-trees rectangles [16].

In this study, we employ a common measure of *severity* to describe the seriousness of atypical data in CPS. The severity function $f(w,t)$ is defined on the spatial and temporal domains, where w can be any region in a spatial coverage W and t can be any time of the temporal range T. Since the CPS sensors are usually fixed in their locations, with the help of a topology graph mapping the sensors to spatial regions, the region W is then represented by a sensor set S that for all $s \in S$, s is located in W. Then the total severity can be computed on discrete sets as (1):

$$F(S,T) = \sum_{s \in S} \sum_{t \in T} f(s,t). \tag{1}$$

Property 2. The total severity $F(S,T)$ is a distributive measure.

Proof. A measure is *distributive* if it can be derived from the aggregation values of n subsets, and the measure is the same as that derived from the entire data set [13].

Let us partition the dataset in S and T into n subsets, each with $S_i \subset S$ and $T_i \subset T$. The severity of the ith subset is computed by aggregating the severities of every subset as shown in the following:

$$F(S_i,T_i) = \sum_{s \in S_i} \sum_{t \in T_i} f(s,t). \tag{2}$$

Then total severity is computed as $F(S,T) = \sum_1^n F(S_i,T_i)$. Since $\bigcup_{i=1,\ldots,n} S_i = S$ and $\bigcup_{i=1,\ldots,n} T_i = T$,

$$F(S,T) = \sum_{s \in S} \sum_{t \in T} f(s,t). \tag{3}$$

Therefore $F(S,T)$ is in the same format from the one derived from the entire dataset, and it is a distributive measure. □

The distributive measure is efficient to compute [13], and the bottom-up styled approach is fast in both cube construction and query processing. However, the information of total severity is too abstract to answer the queries like "where and how do the traffic congestions start and expand?"

Example 6. The bottom-up styled cube is constructed by zipcode areas. The regions with high total severity are tagged out as red zones, for example, $a, b, \ldots, g$ in Figure 2. However the cube only points out where the congestions are. It does not give detailed information on when those congestions start and which part is the most serious in a specified red zone.

The bottom-up styled cube cannot provide details since the numeric measure of total severity is not enough to describe the complex atypical events. In addition, the atypical events may not follow predefined regions. The three major congestions A, B, and C in Figure 2 are partitioned into seven red zones by bottom-up styled cube. It is natural to lead users to illusions that the fragments of A and B congest together in area a. But a careful examination reveals that the highway segments A (freeway 10W) usually congest in the

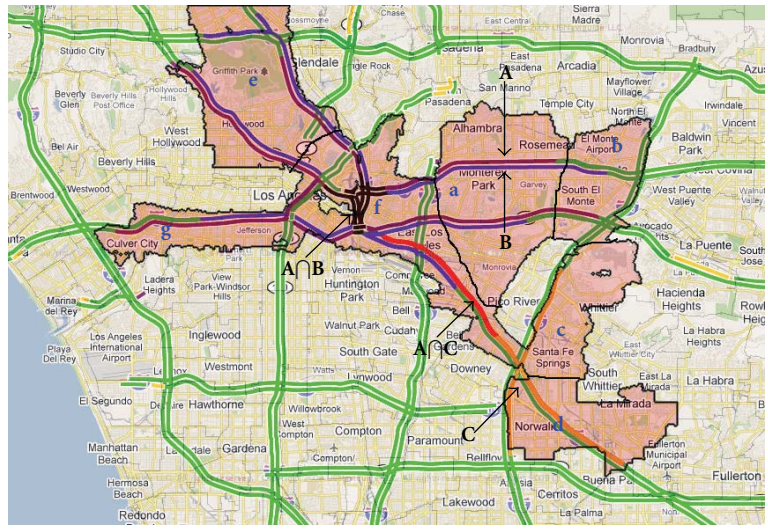


Figure 2: Problems of Bottom-up Styled Cube.

morning rush hours and the segments B (freeway 10E) jam in the evenings. They seldom congest together and should be distinguished from each other.

3.2. Basic Atypical Cluster. In real applications, the users usually cannot provide accurate boundaries to separate atypical events. Instead, the system is required to discover such boundaries automatically and distinguish different atypical events to the users. For this purpose, we propose the concept of *basic atypical cluster*.

Definition 7 (*Basic Atypical Cluster*). Let E be an atypical event with sensor set $S = \{s_1, s_2, \ldots, s_n\}$ and time window sequence $T = \{t_1, t_2, \ldots, t_m\}$; the *basic atypical cluster* C of E is defined as $C = \langle \text{IF}, \text{SF}, \text{TF} \rangle$, in which IF is the identity features, such as cluster ID, date, street name, and highway numbers; the spatial feature $\text{SF} = \{\langle s_1, \mu_1 \rangle, \langle s_2, \mu_2 \rangle, \ldots, \langle s_n, \mu_n \rangle\}$, $\mu_i = \sum_T f(s_i, t)$ is the aggregated severity of sensor s_i; the temporal feature $\text{TF} = \{\langle t_1, \nu_1 \rangle, \langle t_2, \nu_2 \rangle, \ldots, \langle t_m, \nu_m \rangle\}$, $\nu_j = \sum_S f(s, t_j)$ is the aggregated severity of time window t_j.

Intuitively speaking, the spatial feature is the summary of the atypical event in temporal dimension, and the temporal feature is the summary of the event in spatial dimension. μ_i represents how long the sensor s_i is atypical in E, and ν_j reflects how many sensors are atypical during time window t_j of E. In this way the basic atypical cluster C denotes the coverage, time length, and seriousness of corresponding atypical event.

Example 8. Table 3 shows the basic atypical clusters retrieved from Example 6. The ID feature is a general description of the corresponding atypical event; for example, C_A is generated from event E_A which happens in highway 10W on October 30th. The spatial and temporal features are generated by aggregating the atypical records. Note that since the sensors may have different atypical durations in a time window, we still use the accumulated time duration to denote the number of atypical sensors in temporal features. The spatial and temporal features can be directly used to answer the queries in Example 1; for example, the congestion event A starts at around 8:05 am, and the most serious part is the road segment monitored by s_1; it experiences total 182 minutes of congestion in event E_A.

Table 3: Example: basic atypical clusters.

ID features	Spatial features	Temporal features
C_A, highway #10W, Oct 30th	$\langle s_1, 182\text{ min} \rangle$; $\langle s_2, 97\text{ min} \rangle$; $\langle s_3, 33\text{ min} \rangle$; $\langle s_4, 12\text{ min} \rangle; \ldots$	$\langle$8:05 am–8:10 am, 4 min$\rangle$; $\langle$8:10 am–8:15 am, 10 min$\rangle; \ldots$
C_B, highway #10E, Oct 30th	$\langle s_1, 12\text{ min} \rangle$; $\langle s_2, 51\text{ min} \rangle$; $\langle s_3, 34\text{ min} \rangle$; $\langle s_4, 140\text{ min} \rangle; \ldots$	$\langle$6:20 pm–6:25 pm, 7 min$\rangle$; $\langle$6:25 pm–6:30 pm, 13 min$\rangle; \ldots$
C_C, highway #5N, Oct 30th	$\langle s_1, 103\text{ min} \rangle$; $\langle s_2, 75\text{ min} \rangle$; $\langle s_7, 54\text{ min} \rangle$; $\langle s_9, 60\text{ min} \rangle; \ldots$	$\langle$8:20 am–8:25 am, 1 min$\rangle$; $\langle$8:25 am–8:30 am, 15 min$\rangle; \ldots$

The atypical events are retrieved by a single scan of the dataset, and the basic atypical clusters can be generated simultaneously. Algorithm 1 shows the detailed process.

The basic cluster generation algorithm randomly picks a seed record from the dataset (Line 2) and retrieves all the atypical connected records to it (Line 3). Then the algorithm groups those records as an atypical event (Line 4) and generates the spatial and temporal features of basic cluster (Lines 6–13). Those steps are repeated until all the data are processed.

```
     Input: the time interval threshold δ_t, distance threshold δ_d, atypical
            dataset R
     Output: The basic atypical cluster set basic_set
1    repeat
2        randomly select a seed record r from R;
3        retrieve all the atypical connected records from r w.r.t. δ_t and δ_d;
4        group those records to form an atypical event E(S; T);
5        initialize basic cluster C⟨IF; SF; TF⟩;
6        foreach sensor s_i ∈ S do
7            compute the sensor severity μ_i;
8            add ⟨s_i, μ_i⟩ to SF;
9        end
10       foreach time window t_j ∈ T do
11           compute the time window severity ν_j;
12           add ⟨t_j, ν_j⟩ to TF;
13       end
14       add C to basic_set;
15       R ← R − E
16   until R = φ;
17   return basic_set;
```

Algorithm 1: Basic cluster generation.

Proposition 9. *The time complexity of Algorithm 1 is $O(n^2)$ without index and $O(n \cdot \log(n))$ with spatial index (e.g., R-tree), where n is the number of atypical records.*

Proof. The major cost of Algorithm 1 is on Line 3 to retrieve the atypical connected records. If there is no index on the temporal and spatial dimensions, it costs $O(n)$ time to retrieve the neighbors of one seed, and each atypical record's connected records are retrieved only once. Thus the entire step takes $O(n^2)$ time. However the neighbor searching algorithm can speed up to $O(\log(n))$ with a spatial index such as R-tree, and hence the time complexity of the algorithm is improved as $O(n \cdot \log(n))$. □

3.3. Atypical Cluster Aggregation. The first task of cluster aggregation is to compute the similarities between two atypical clusters. The cluster similarity is measured based on both the spatial and temporal features, as shown in (4). Two clusters are considered similar to each other only when they have atypical records at the same places during the same time periods. Equations (5) and (6) show the calculation of spatial and temporal similarities, where S_i is the sensor set, T_i is the time window set, and μ^i and ν^i are the aggregated severity of sensor s and time window t in cluster C_i, respectively. Equation (5) computes the severity percentages of common sensors over a cluster and balances the values on two clusters by a mathematical function $g(p_1, p_2)$. The function $g(p_1, p_2)$ could be in the form of max, min, the arithmetic mean, harmonic mean, or geometric mean. The reason of using different mathematical balance function here is that the size of two clusters may be different. When comparing the similarity between a large cluster and a small one, the percentage of common sensors is inevitably small for the larger cluster. If we use the max function, the two clusters are still similar even if the common sensor percentage is low for the larger cluster:

$$\mathrm{Sim}(C_1, C_2) = \frac{1}{2}(\mathrm{Sim}_{\mathrm{SF}}(C_1, C_2) + \mathrm{Sim}_{\mathrm{TF}}(C_1, C_2)), \tag{4}$$

$$\mathrm{Sim}_{\mathrm{SF}}(C_1, C_2) = g\left(\frac{\sum_{S_1 \cap S_2} \mu^1}{\sum_{S_1} \mu^1}, \frac{\sum_{S_1 \cap S_2} \mu^2}{\sum_{S_2} \mu^2}\right), \tag{5}$$

$$\mathrm{Sim}_{\mathrm{TF}}(C_1, C_2) = g\left(\frac{\sum_{T_1 \cap T_2} \nu^1}{\sum_{T_1} \nu^1}, \frac{\sum_{T_1 \cap T_2} \nu^2}{\sum_{T_2} \nu^2}\right). \tag{6}$$

Once two micro-clusters are merged, a single macro-cluster is created to represent the result out of this merge (here we use the term *micro-cluster* to denote the merge input and *macro-cluster* to denote the merge result). The spatial feature of the macro-cluster is calculated as shown in (7): the system accumulates the severities of common sensors from two micro-clusters and keeps the nonoverlapping ones; so is the temporal feature. A new ID is generated for the macro-cluster:

$$\begin{aligned} \mathrm{SF}_{\mathrm{new}} = {} & \left\{\left\langle s_i, \mu_i^1 + \mu_i^2 \right\rangle \mid s_i \in (S_1 \cap S_2)\right\} \\ & \cup \left\{\left\langle s_j, \mu_j \right\rangle \mid s_j \notin (S_1 \cap S_2), s_j \in (S_1 \cup S_2)\right\}. \end{aligned} \tag{7}$$

Property 3. The spatial and temporal features in atypical clusters are algebraic measures.

Proof. A measure is *algebraic* if it can be computed by an algebraic function with m arguments (m is a bounded positive integer), and each of the arguments is distributive [13]. We will prove that the spatial feature is algebraic in the process of integrating n micro-clusters to a macro-cluster by mathematical induction.

(1) *The Basis*. First we study the case that $n = 2$.

Let S_1 and S_2 be the sensor sets of two micro-clusters; the spatial feature of macro-cluster $\mathrm{SF}_{\mathrm{macro}}$ is computed as (8):

$$\begin{aligned}\mathrm{SF}_{\mathrm{macro}} = {} & \{\langle s_i, \mu_i^1 + \mu_i^2\rangle \mid s_i \in (S_1 \cap S_2)\} \\ & \cup \left\{\left\langle s_j, \mu_j\right\rangle \mid s_j \notin (S_1 \cap S_2), s_j \in (S_1 \cup S_2)\right\}.\end{aligned} \tag{8}$$

The sensor severity μ is a distributive measure, according to the definition, and spatial feature is algebraic when $n = 2$.

(2) *The Inductive Step.* Suppose that the statement holds for $n - 1$; we study the case of integrating n micro-clusters.

The macro-cluster C_N can be seen as the integration of the macro-cluster C_{N-1} and the nth micro-cluster C_n. Let S_{N-1} and S_n be the corresponding sensor sets; the spatial feature of macro-cluster SF_N is computed as (9):

$$\begin{aligned}\mathrm{SF}_N = {} & \{\langle s_i, \mu_i^1 + \mu_i^2\rangle \mid s_i \in (S_{N-1} \cap S_n)\} \\ & \cup \left\{\left\langle s_j, \mu_j\right\rangle \mid s_j \notin (S_{N-1} \cap S_n), s_j \in (S_{N-1} \cup S_n)\right\}.\end{aligned} \tag{9}$$

Therefore the statement holds for case of integrating n micro-clusters. The spatial feature is algebraic.

From the same steps, it is easy to obtain that the temporal feature is also algebraic. □

The algebraic measures are also efficient to compute and aggregate [13]; thus we use atypical clusters as the measure in atypical cube. The detailed micro-cluster merging steps are shown in Algorithm 2. The system accumulates the severity of common sensors in two micro-clusters (Lines 2–6) and copies the nonoverlapping ones (Line 7); the same steps are carried out in temporal features (Lines 9–16).

Property 4. The operation of merging atypical clusters is mathematically commutative and associative.

Proof. To prove that the merge operation is mathematically commutative, we have to show that for any C_1 and C_2, C_1 merge $C_2 = C_2$ merge C_1.

For two clusters C_1 and C_2, the spatial feature of their integrating cluster is $\mathrm{SF}_{\mathrm{new}}$ computed as

$$\begin{aligned}\mathrm{SF}_{\mathrm{new}} = {} & \{\langle s_i, \mu_i^1 + \mu_i^2\rangle \mid s_i \in (S_1 \cap S_2)\} \\ & \cup \left\{\left\langle s_j, \mu_j\right\rangle \mid s_j \notin (S_1 \cap S_2), s_j \in (S_1 \cup S_2)\right\}.\end{aligned} \tag{10}$$

The positions of S_1 and S_2 are equal in the above equation; $\mathrm{SF}_{\mathrm{new}}$ is not influenced by the order of C_1 and C_2. It is the same for temporal feature computation. And the identity feature is generated independently. Therefore for any C_1 and C_2, C_1 merge $C_2 = C_2$ merge C_1. The merge operation is mathematical commutative.

To prove that the merge operation is mathematically associative, we have to show that for any C_1, C_2 and C_3, (C_1 merge C_2) merge $C_3 = C_1$ merge (C_2 merge C_3).

Let us denote the following:

$$C_4 = C_1 \text{ merge } C_2;\ C_5 = C_2 \text{ merge } C_3;$$

$$C_6 = C_4 \text{ merge } C_3 = (C_1 \text{ merge } C_2) \text{ merge } C_3;$$

$$C_7 = C_1 \text{ merge } C_5 = C_1 \text{ merge } (C_2 \text{ merge } C_3).$$

The spatial feature $\mathrm{SF}(C_6)$ is computed as

$$\begin{aligned}\mathrm{SF}(C_6) = {} & \left\{\left\langle s_i, \mu_i^4 + \mu_i^3\right\rangle \mid s_i \in (S_4 \cap S_3)\right\} \\ & \cup \{\langle s_i, \mu_i\rangle \mid s_i \notin (S_4 \cap S_3), s_i \in (S_4 \cup S_3)\}.\end{aligned} \tag{11}$$

Since $S_4 = S_1 \cup S_2$, (11) can be written as

$$\begin{aligned}\mathrm{SF}(C_6) = {} & \left\{\left\langle s_i, \mu_i^{1,2} + \mu_i^3\right\rangle \mid s_i \in ((S_1 \cup S_2) \cap S_3)\right\} \\ & \cup \{\langle s_i, \mu_i\rangle \mid s_i \notin ((S_1 \cup S_2) \cap S_3), \\ & \qquad s_i \in ((S_1 \cup S_2) \cup S_3)\} \\ = {} & \left\{\left\langle s_i, \mu_i^1 + \mu_i^2 + \mu_i^3\right\rangle \mid s_i \in (S_1 \cap S_2 \cap S_3)\right\} \\ & \cup \left\{\left\langle s_i, \mu_i^1 + \mu_i^3\right\rangle \mid s_i \in (S_1 \cap S_3), s_i \notin S_2\right\} \\ & \cup \left\{\left\langle s_i, \mu_i^2 + \mu_i^3\right\rangle \mid s_i \in (S_2 \cap S_3), s_i \notin S_1\right\} \\ & \cup \{\langle s_i, \mu_i^1 + \mu_i^2\rangle \mid s_i \in (S_1 \cap S_2), s_i \notin S_3\} \\ & \cup \{\langle s_i, \mu_i\rangle \mid s_i \in (S_1 \cup S_2 \cup S_3), \\ & \qquad s_i \notin (S_1 \cap S_3), s_i \notin (S_2 \cap S_3), s_i \notin (S_1 \cap S_2)\}.\end{aligned} \tag{12}$$

Since $S_5 = S_2 \cup S_3$, (12) can be converted to

$$\begin{aligned}\mathrm{SF}(C_6) = {} & \left\{\left\langle s_i, \mu_i^1 + \mu_i^{2,3}\right\rangle \mid s_i \in (S_1 \cap (S_2 \cup S_3))\right\} \\ & \cup \{\langle s_i, \mu_i\rangle \mid s_i \notin (S_1 \cap (S_2 \cup S_3)), \\ & \qquad s_i \in (S_1 \cup (S_2 \cup S_3))\} \\ = {} & \left\{\left\langle s_i, \mu_i^1 + \mu_i^5\right\rangle \mid s_i \in (S_1 \cap S_5)\right\} \\ & \cup \{\langle s_i, \mu_i\rangle \mid s_i \notin (S_1 \cap S_5), s_i \in (S_1 \cup S_5)\} \\ = {} & \mathrm{SF}(C_7).\end{aligned} \tag{13}$$

Equation (13) shows that the spatial features are the same for the macroclusters C_6 and C_7, so are the temporal features. And the identity feature is generated independently. Hence the merge operation is mathematical associative. □

Property 4 tells us that the order of micro-clusters does not influence the macro-cluster results. Thus we design the aggregation clustering process as Algorithm 3. The algorithm starts by checking each pair of the micro-clusters. If their similarity is larger than the given threshold, a merge operation is called to integrate them (Lines 2–4). This process is irrelevant to the order of micro-clusters. The new cluster is put back to the set, and the old pair is discarded (Lines 5–6). The program stops until no clusters could be merged (Line 9).

Proposition 10. *Let m be the number of micro-clusters; the time complexity of Algorithm 3 is $O(m^2)$.*

Proof. In the worst case, there are no similar pairs that can be found to be merged together. Then the algorithm needs to check the similarity between every pair and the total calculation times are $m(m-1)/2$. Hence the time complexity of Algorithm 3 is $O(m^2)$. □

Input: micro-clusters $C_1\langle \mathrm{IF}_1, \mathrm{SF}_1, \mathrm{TF}_1\rangle$ and $C_2\langle \mathrm{IF}_2, \mathrm{SF}_2, \mathrm{TF}_2\rangle$
Output: macro-cluster $C_{\mathrm{new}}\langle \mathrm{IF}_{\mathrm{new}}, \mathrm{SF}_{\mathrm{new}}, \mathrm{TF}_{\mathrm{new}}\rangle$

```
1  foreach sensors_i ∈ SF_1 do
2      if s_i ∈ SF_2 then
3          μ_i^new = μ_i^1 + μ_i^2;
4          add ⟨s_i, μ_i^new⟩ to SF_new;
5          remove the records of s_i from SF_1 and SF_2;
6      end
7      add the rest records of SF_1, SF_2 to SF_new
8  end
9  foreach time windowt_j ∈ TF_1 do
10     if t_j ∈ TF_2 then
11         ν_j^new = ν_j^1 + ν_j^2;
12         add ⟨s_j, μ_j^new⟩ to TF_new;
13         remove the records of t_j from TF_1 and TF_2;
14     end
15     add the rest records of TF_1, TF_2 to TF_new;
16 end
17 generate IF_new;
18 return C_new;
```

Algorithm 2: Merge Micro-Clusters.

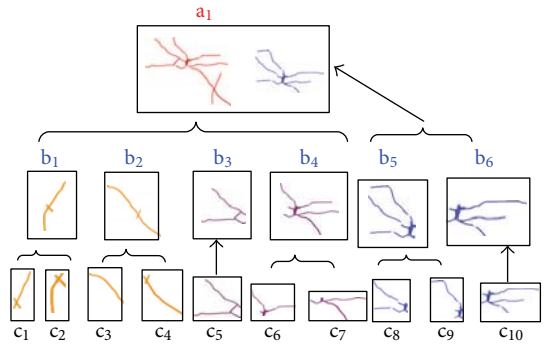


Figure 3: Example: the framework of atypical cube.

Note that the similarity threshold δ_{sim} is an important parameter for Algorithm 3. δ_{sim} should be set larger than 0.5, since the clusters should be both spatially close and temporally related. On the other hand, if δ_{sim} is too high (e.g., $\delta_{\mathrm{sim}} = 1$), no atypical cluster will merge with other ones and the macroclusters with high severity cannot be generated. We have conducted a performance study in Section 5.3; the experiment results suggest that the algorithm has the best performance if δ_{sim} is set around 0.6.

Example 11. Table 3 lists three atypical clusters, namely, C_A, C_B, and C_C. Suppose that δ_{sim} is set as 0.6. The aggregation clustering algorithm first checks the cluster pair of C_A and C_B and computes their similarity. Since they are only spatially close but not timely related, $\mathrm{Sim}(C_A, C_B) = 0.49$, which is less than δ_{sim}. Hence this pair will not be integrated in one cluster. Then, the system picks the pair of C_A and C_C, since these two clusters have most of their atypical data in the common areas with similar times, $\mathrm{Sim}(C_A, C_C) = 0.87$; thus they should be merged according to (4). The merged result is tagged as cluster C_D. And the similarity between C_B and C_D is still lower than δ_{sim}. The aggregation clustering algorithm stops and outputs C_B and C_D as the results.

The aggregation clustering algorithm takes the micro-clusters from children cells as input and outputs the macroclusters to store as measures in the parent cell. Such macroclusters are also going to be used as new inputs to get even higher-level clusters. In this way a hierarchical clustering tree is built up.

The atypical cube is constructed as a forest of hierarchical clustering trees, where each tree represents an aggregation path in the cube. Figure 3 shows the framework of atypical cube for the case of traffic congestions. Ten basic atypical clusters are stored in the lowest-level cells. The aggregation cells $b_1, \ldots, b_6$ are built on them, and the apex cell a_1 has two macroclusters integrated from the micro-clusters in $b_1, \ldots, b_6$.

4. OLAP Query Processing

4.1. Processing Large-Scale Queries. In practical applications we do not precompute the entire atypical cube due to storage limits. In most cases only the basic clusters and some low-level micro-clusters are precomputed. With such a partially materialized data structure, the system needs to dynamically integrate the low-level clusters to process OLAP queries in large query range. The online clustering process is similar to the cluster integration algorithm. However there are two

problems. (1) The first one is efficiency: the time complexity of cluster aggregation algorithm is quadratic to the number of input clusters. Therefore the system should select only the relevant micro-clusters to reduce the time cost. (2) The second is effectiveness: if the query scale is large, for example, the users want the monthly congestion report of the whole city. There may be a large number of macroclusters in the query range, but only few of them are *significant clusters* with high severities, while the others are negligible. When constructing atypical cube, the process is offline and the system can store all the clustering results. When processing online queries, the users usually demand the significant clusters being delivered in short time and do not prefer the results mixed with trivial clusters.

Definition 12 (*Significant Cluster*). Let $Q(W, T)$ be a query with the range in region W and time T. The cluster C is significant if severity$(C) > \delta_s \cdot \text{length}(T) \cdot N$, where δ_s is the severity threshold and N is the number of sensors in W and severity$(C) = \sum_{\text{SF}} \mu_i = \sum_{\text{TF}} \nu_j$.

Note that the system measures the cluster significance by a relative threshold δ_s, because severity(C) is influenced by the query scales; for example, the severities of high-level clusters in one month are usually larger than the low-level clusters in a day.

The key challenge for online clustering is to prune the trivial micro-clusters and meanwhile guarantee the accuracy of significant macroclusters. One strategy is beforehand pruning: the system pushes down the prune step to lower levels by only selecting the significant micro-clusters for integration. However this strategy cannot guarantee finding all the significant macroclusters, because a micro-cluster that contributes to a significant macro-cluster may not be significant by itself. If the algorithm prunes all insignificant micro-clusters beforehand, the severity of the macro-cluster will also be reduced and may not be significant anymore.

Example 13. The micro-clusters of Los Angeles in October 30th are shown in Figure 4(a), and the monthly significant macroclusters A and B are plotted in Figure 4(b). In Figure 4(a), the micro-clusters a, b, j, k, and o are going to be integrated as parts of the significant macroclusters even if they are relatively trivial. The micro-clusters e, h, and i are significant in the scale of one day, but actually they can be pruned since they have no contribution for any significant macroclusters in one month.

Can We Foretell Which Microcluster Will Become a Part of the Significant Macroclusters and Which Will Not? If the system knows such guiding information, it can improve query efficiency and meanwhile guarantee the result's accuracy. The heuristic comes from the bottom-up method: recall that the bottom-up method uses total severity $F(S, T)$ as the measure. As a distributive measure, $F(S, T)$ is efficient to compute [13] and can be employed as the guidance to retrieve significant clusters.

One may worry that $F(S, T)$ is computed on predefined regions such as zipcode areas, and their boundaries are different from the atypical clusters. Fortunately, Property 5 shows that there is a relation between predefined regions and atypical clusters.

Property 5. Let $Q(W, T)$ be an OLAP query, let δ_s be the relative severity threshold, let W' be a spatial region that $W' \subseteq W$, and let S' and S be the sensor sets installed in regions $W'$and W, respectively. If $F(S', T) < \delta_s \cdot \text{Length}(T) \cdot |S|$, then there is no significant macro-cluster in S' within time T.

Proof. We will prove the statement by contradiction.

Suppose that there is a cluster C_j with sensor set $S_j \subseteq S'$ and time window sequence $T_j \subseteq T$, such that Severity$(C_j) \geq \delta_s \cdot \text{Length}(T) \cdot |S|$.

Since $F(S', T)$ is the aggregation of total severity in S' and T,

$$F(S', T) \geq \text{Severity}\left(C_j\right) \geq \delta_s \cdot \text{Length}(T) \cdot |S|. \quad (14)$$

We now have a contradiction with the condition that $F(S', T) < \delta_s \cdot \text{Length}(T) \cdot |S|$. Hence there does not exist such cluster C_j. □

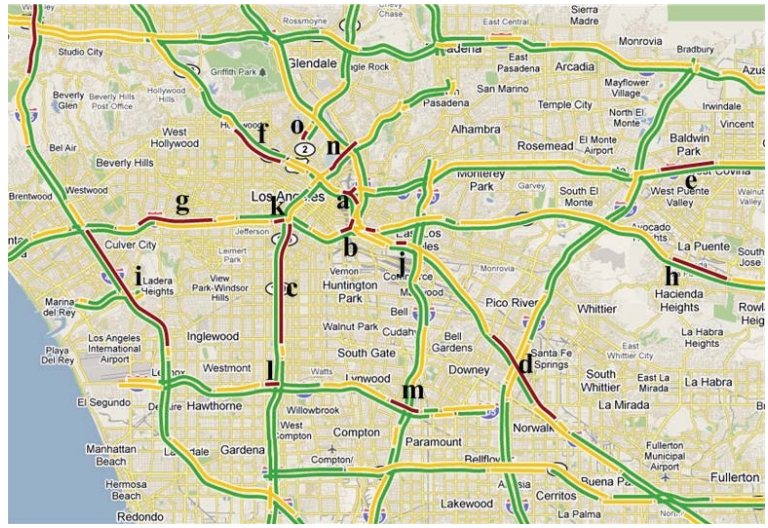
Property 5 can be used to help filtering the micro-clusters. The system only needs to integrate the clusters in the regions where the total severities are larger than threshold, that is, the *red zones*.

Example 14. In Figure 5, the red zones are tagged out. They are generated by the bottom-up styled cube with a predefined zipcode area hierarchy. The micro-clusters e, g, i, and m can be pruned safely since they are outside the zones; a, b, and d should be kept for clustering since they are in the zones; c, k, f, o, and n are also kept since they intersect with the red zones and may contribute to macroclusters.

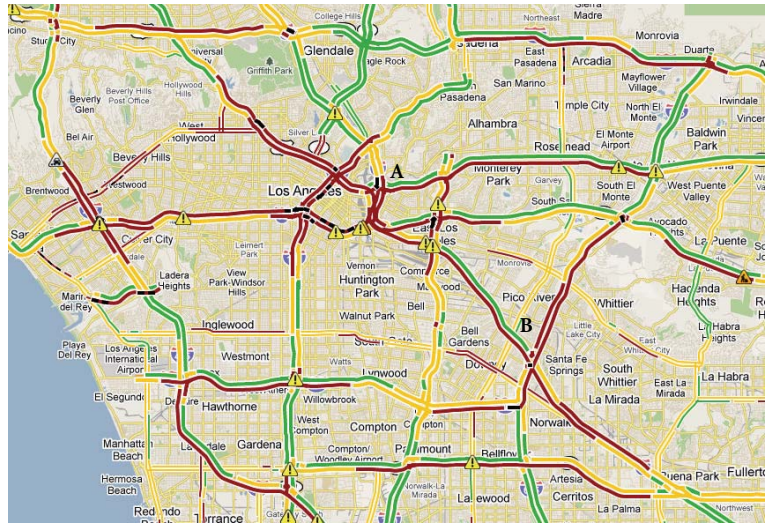
Algorithm 4 shows detailed steps of red zone guided online clustering. The system first computes the severity on predefined regions in bottom-up styled cube and retrieves the red zones (Lines 1–4) and then selects out the micro-clusters in red zones (Lines 5–7). The clustering algorithm (Algorithm 1) is called to generate the macroclusters (Line 8). Since the algorithm can only guarantee there are no false negatives (i.e., not missing any significant macroclusters), it is possible to generate some false positives. A check procedure is processed to prune the clusters without enough severity at the last step (Lines 9–11).

The major cost of Algorithm 4 is at Line 8 to call the clustering algorithm. In the worst case, no cluster could be filtered out, and the algorithm's time complexity is still quadratic to the number of micro-clusters. However, in our experiments, about 80% micro-clusters could be filtered out with reasonable δ_s, and the query efficiency was improved dramatically.

4.2. Drill-Through Query Processing. In some rare cases, the users require detailed results in a very narrow range, such as "what is the congestion details from 8:45 am to 9 am of the highway #101 near downtown?" The system needs



(a) Microclusters in October 30th



(b) Significant macroclusters in October

Figure 4: Example: problem of beforehand pruning.

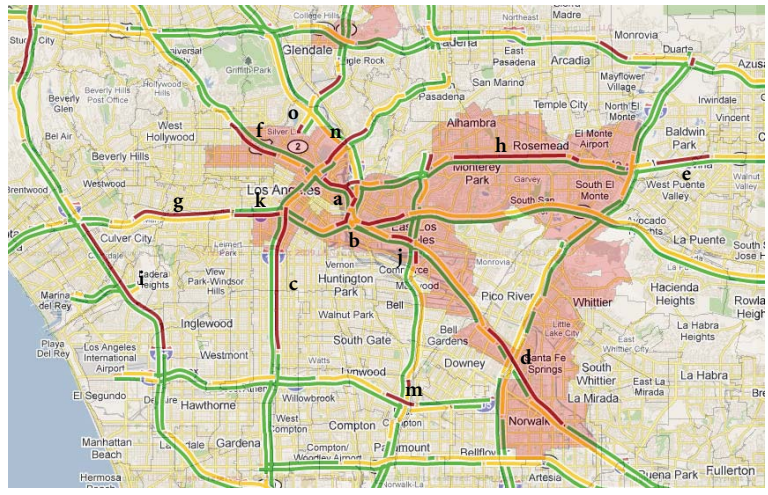


Figure 5: Red-zone guided clustering.

Input: micro-cluster set *micro_set* from lower cells, similarity threshold δ_{sim}
Output: the macro-cluster set *macro_set*

```
1  repeat
2      foreach micro-cluster pair C_i, C_j do
3          if Sim(C_i, C_j) > δ_sim then
4              C_new = merge(C_i, C_j);
5              add C_new to micro_set;
6              remove C_i, C_j from micro_set;
7          end
8      end
9  until no clusters can be merged in micro_set;
10 macro_set ← micro_set;
11 return macro_set;
```

Algorithm 3: Aggregation clustering.

Input: query $Q(S, T)$, micro-cluster set *micro_set* from materialized children cells, bottom-up style cube *bu_cube*, similarity threshold δ_{sim}, relative severity threshold δ_s
Output: the significant macro-cluster set *sig_set* for the query

```
1  Compute F(S_i, T) from bu_cube for Q(S, T);
2  If F(S_i, T) ≥ δ_s · Length(T) · |S| then
3      add S_i to red zone set red_set;
4  end
5  foreach micro-cluster C_i ∈ micro_set do
6      if C_i ∈ red_set or C_i intersects with red_set then add C_i to
7      sig_micro_set;
7  end
8  sig_set ← clustering (sig_micro_set, δ_sim) (Algorithm 1)
9  foreach cluster C_i ∈ sig_set do
10     if Severity(C_i) < δ_s · Length(T) · |S| then remove C_i;
11 end
12 return sig_set;
```

Algorithm 4: Red-zone guided clustering.

to decompose basic atypical clusters to answer them. Such queries actually *drill through* the atypical cube to the original sensor dataset.

We propose a two-stage algorithm to process this kind of queries. The system first returns the related basic atypical clusters as an approximate result and then decomposes them for precise answers. The details are shown in Algorithm 5 as a two-stage process: in the first stage, the system retrieves all related micro-clusters, directly integrates them and returns the macroclusters as approximate result (Lines 1–5); then it drills through to the sensor dataset and refines those micro-clusters in the second stage (Lines 6–12). The precise results are computed and returned later.

Since the query range is narrow, the size of the *micro_set* is usually small; the major cost of Algorithm 5 is in the drill-through step with high I/O overhead (Line 7). However if the system builds an index for the atypical events in the sensor data and maintains an inverted pointer p from each basic atypical cluster to the corresponding atypical events, the time cost of Algorithm 5 will be improved significantly.

5. Performance Evaluation

Since the idea of this study is motivated by practical application problems, we use real world datasets to evaluate the proposed approaches. Twelve datasets are collected from the PeMS traffic monitoring system [11]; each stores one-month traffic data in the areas of Los Angeles and Ventura. The data are collected from over 4,000 sensors on 38 highways. There are more than 1.1 million records for a single day and totally 428 million records for the whole year. The total size of all the datasets is over 54 GB.

The experiments are conducted on a PC with an Intel 2200 Dual CPU at 2.20 G Hz and 2.19 G Hz. The RAM is 1.98 GB, and the operating system is Windows XP SP2. All the algorithms are implemented in Java on Eclipse 3.3.1

Input: atypical cube *cube*, drill-through query Q, similarity threshold δ_{sim}
Output: The approximate result R_a, the precise result R_p

```
1   foreach basic atypical cluster C_i in cube do
2       if C_i related to Q then add C_i to micro_set;
3   end
4   R_a ← clustering (micro_set, δ_sim) (Algorithm 1);
5   return R_a;
6   foreach C_i do
7       drill through to the sensor dataset;
8       C_i' ← filter the records in C_i with Q's condition;
9       add C_i' to micro_set';
10  end
11  R_p ← clustering (micro_set', δ_sim);
12  return R_p;
```

Algorithm 5: Drill-through query processing.

Table 4: Experiment settings and parameters.

Dataset	Date	Sensor. no.	Reading. no.	Atypical Data %
D_1	Oct. 2008	4,076	$3.4*10^7$	~2.3%
D_2	Nov. 2008	4,052	$3.3*10^7$	~3.7%
...	...	...	...	...
D_{12}	Sep. 2009	4,076	$3.3*10^7$	~4.0%

the severity threshold δ_s: 2%–20%, default 5%
the distance threshold δ_d: 1.5 mile–24 mile, default 1.5 mile
the time interval threshold δ_t: 15 min–80 min, default 15 min
the similarity threshold δ_{sim}: 0.1–1, default 0.5
the g function: max, min, arithmetic mean, harmonic mean, and geometric mean, default: arithmetic mean

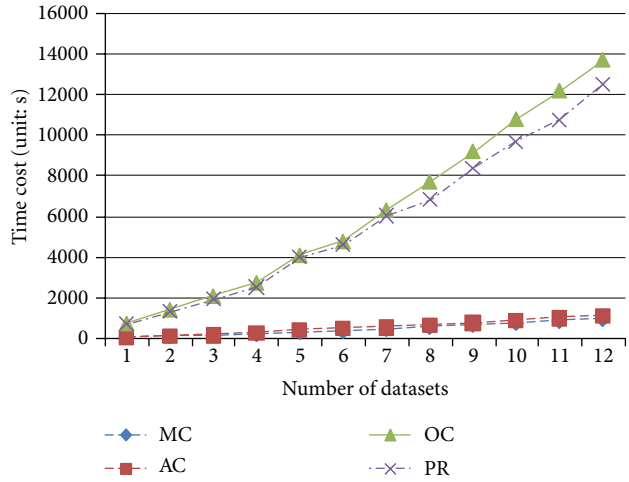


Figure 6: Efficiency: construction time cost versus no. of datasets.

platform with JDK 1.5.0. The detailed experimental settings and parameters are listed in Table 4.

5.1. Evaluations of Cube Construction. In this subsection we evaluate the algorithms of offline cube construction. *CubeView* [14] is a bottom-up method on traffic data. The original CubeView algorithm aggregates all the traffic records with predefined spatial and temporal hierarchies. In this experiment, the system carries out a preprocessing step to select atypical records and adjusts CubeView to construct the cube only on the atypical data. We first construct the cubes on a single dataset, then gradually increase the number of datasets, until all the twelve datasets are used in the experiment. Figure 6 shows the time costs of the original Cubeview (OC), modified CubeView (MC), the preprocessing step (PR), and our atypical-cluster-based method (AC). The x-axis is the number of datasets used in the experiment, and y-axis is the time cost of the algorithms. MC and AC are an order of magnitude faster than OC because they are constructed on the atypical data, which are only 2% to 5% of the original datasets. The time cost of PR is close to OC since both of them have to scan the original datasets with huge I/O overhead. However the preprocessing step only needs to carry out once for constructing different cubes.

Figure 7 shows the constructed cube size of original Cubeview (OC), modified CubeView (MC), and atypical cluster method (AC). The size of corresponding atypical events (AEs) is also recorded in the figure. MC achieves the best compression effects since it only records the numeric measure of total severity, but it cannot describe the complex atypical events. AC stores all the critical information about spatial and temporal features of AE, but only costs 0.5% to 1% space of AE.

Figure 8(a) records the constructed cube size of the three methods on six different datasets. In general, the construction overhead and storage size of OC are acceptable since the cube is built offline. However, the results in Figure 8(b) show that OC cannot be used to process queries about atypical events. The average speeds of OC on all the datasets are all around 65 mph, which are close to the speed limit of California highway. The information is hence not

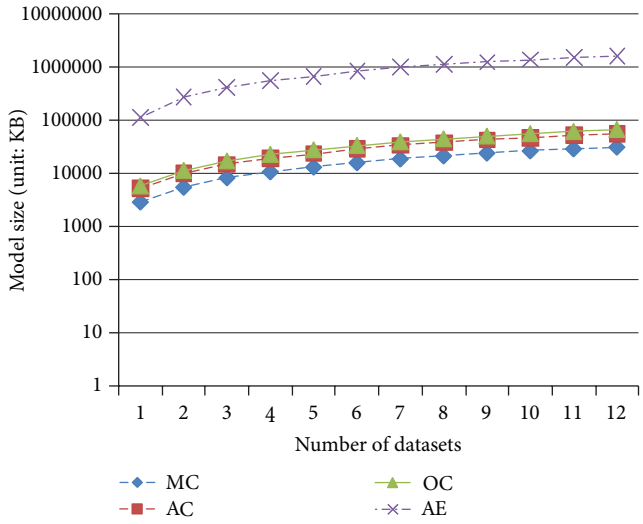


Figure 7: Size: constructed cube size versus no. of datasets.

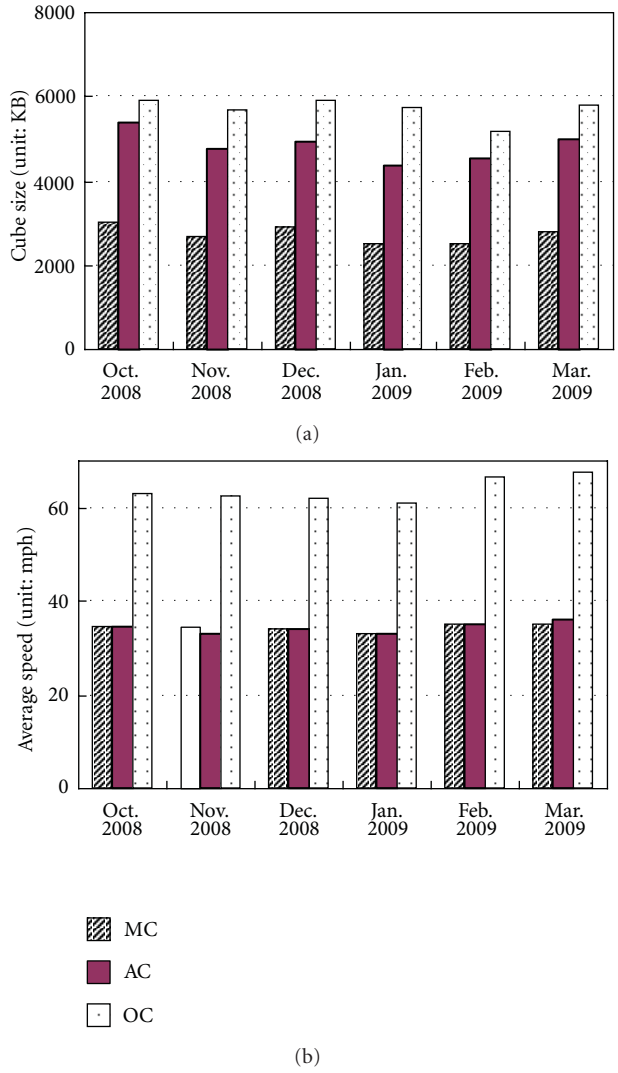


Figure 8: Results: cube size and avg. speed.

interesting to the users since the atypical events are dwarfed by the majorities of normal data.

5.2. Comparisons in OLAP Query Processing. In this subsection we evaluate the performances of OLAP query processing. Three query processing strategies are compared: (1) integrating all the micro-clusters (All); (2) pruning the insignificant clusters beforehand (Pru); (3) the red-zone guided clustering (Gui).

In the experiments, the system only precomputes the micro-clusters of each day. The analytical query's spatial range is fixed as Los Angeles, and time range gradually increases from one week (requiring to aggregate the micro-clusters of 7 days) to three months (84 days). Figures 9(a) and 9(b) record the average time and I/O costs (measured by the number of input micro-clusters). Although Gui has extra cost to compute the redzones, the time efficiency is still close to Pru. From the figure one can see clearly that Gui and Pru are much more efficient than All. Gui's time cost is only about 15%–20% of All.

To evaluate the effectiveness of query results, we compute the precision and recall of the significant clusters. Since the integrating-all method prunes no clusters, its results contain all the significant clusters. The system checks the results of All and retrieves the true significant clusters as the ground truth. The measures of precision and recall are then computed as follows.

Precision. The precision is calculated as the proportion of significant clusters in the returned query results.

Recall. The recall is the proportion of retrieved significant clusters over the ground truth.

The system increases the query time range from 7 days to 84 days and records the precision and recall of three methods in Figure 10. For all the methods, their precision decreases with larger query range, because the cluster severity does not grow linearly with respect to the query range, and the significant clusters are inevitably fewer in larger query range with fixed severity threshold. In the experiment, Pru has the highest precision, because it prunes all the trivial micro-clusters and generates fewer macroclusters (Figure 10(a)). However, as shown in Figure 10(b), Pru cannot guarantee to find all the significant clusters. Its recall might even be lower than 50% in some cases. Therefore, even if Pru is the winner on efficiency and precision, it is not feasible to process analytical query since the significant clusters may be missed in the results.

In the next experiment, we fix the query time range as 14 days and evaluate the influence of severity threshold δ_s. The experimental results are shown in Figure 11. The precision drops with larger δ_s, because fewer macroclusters can meet the high standard of severity to become significant. Another interesting observation is that the recall of Pru increases

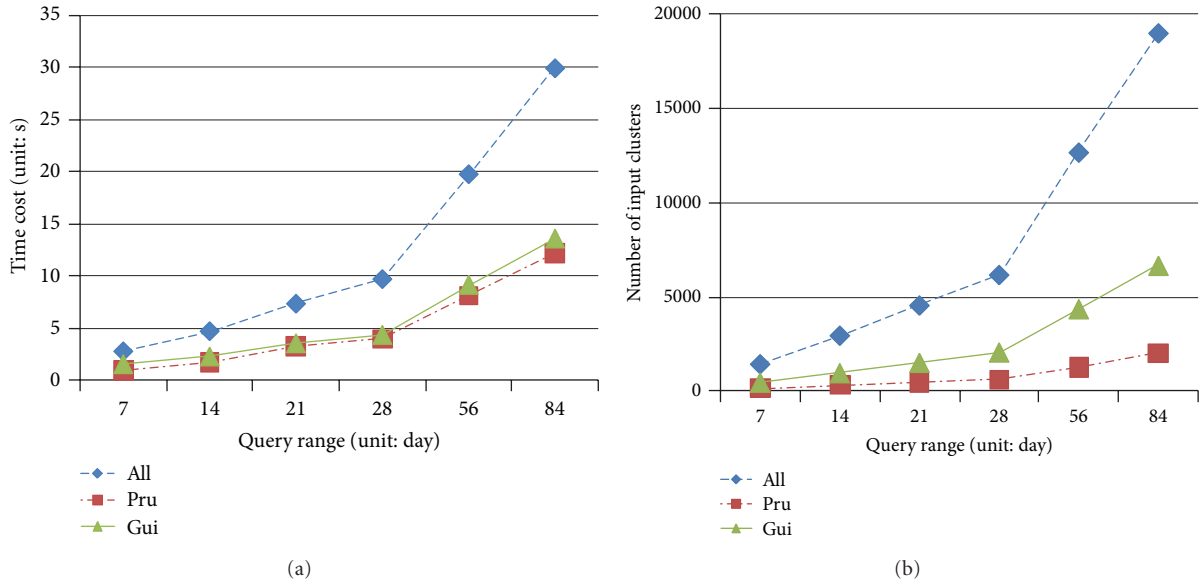


Figure 9: Efficiency: query time and I/O costs.

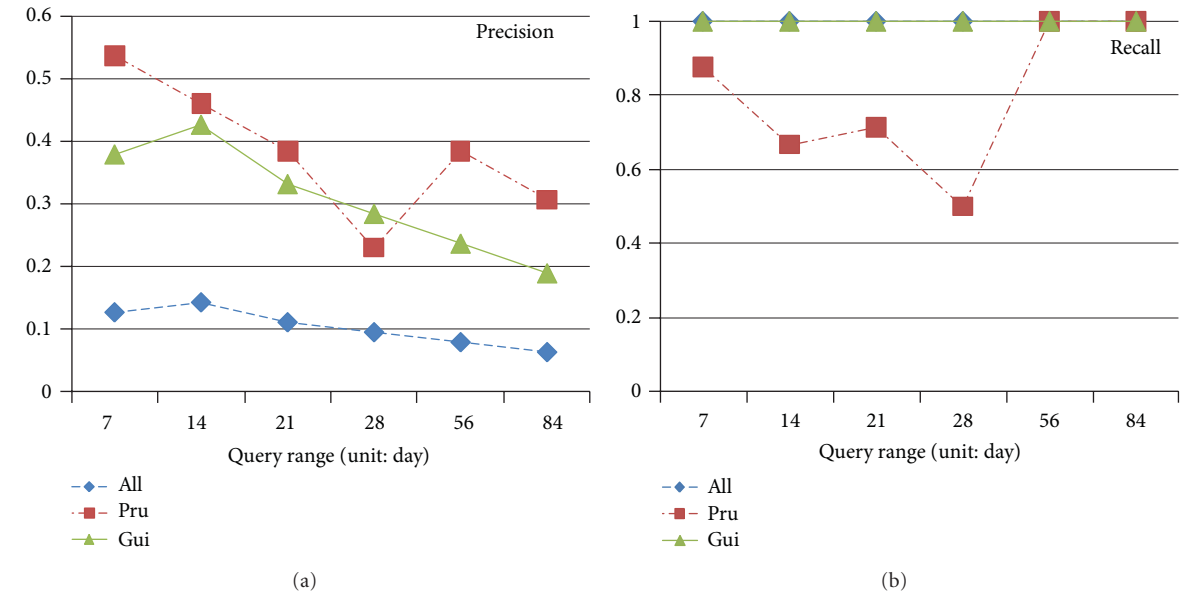


Figure 10: Effectiveness: precision and recall with respect to query range.

when δ_s grows. Pru is unlikely to miss the macroclusters with very high severities. However, the detailed spatial and temporal features of those clusters may not be accurate, because Pru filters out some micro-clusters that should be integrated in.

The precision of Gui in the above experiments is not high; however this measure can be easily improved. The system can efficiently filter out the false positives and guarantee 100% precision by checking the macro-cluster's total severity. Gui has such a procedure (Lines 5–7 in Algorithm 4). This procedure is turned off in the experiments for a fair play.

5.3. Parameter Tuning for Atypical Cluster Method. In the next experiment, we study the influence of the parameters in the atypical-cluster-based method, including time interval threshold δ_t, distance threshold δ_d, similarity threshold δ_{sim}, and balance function g. The system first retrieves the micro-clusters in each day with different δ_t and δ_d and then carries out the cluster integration to generate the macroclusters for every week and month. Figure 12(a) shows the average number of the atypical clusters in every day, week, and month. The figure also records the average number of weekly/monthly significant clusters as sig(week)/sig(month). One can clearly see that the numbers of weekly and

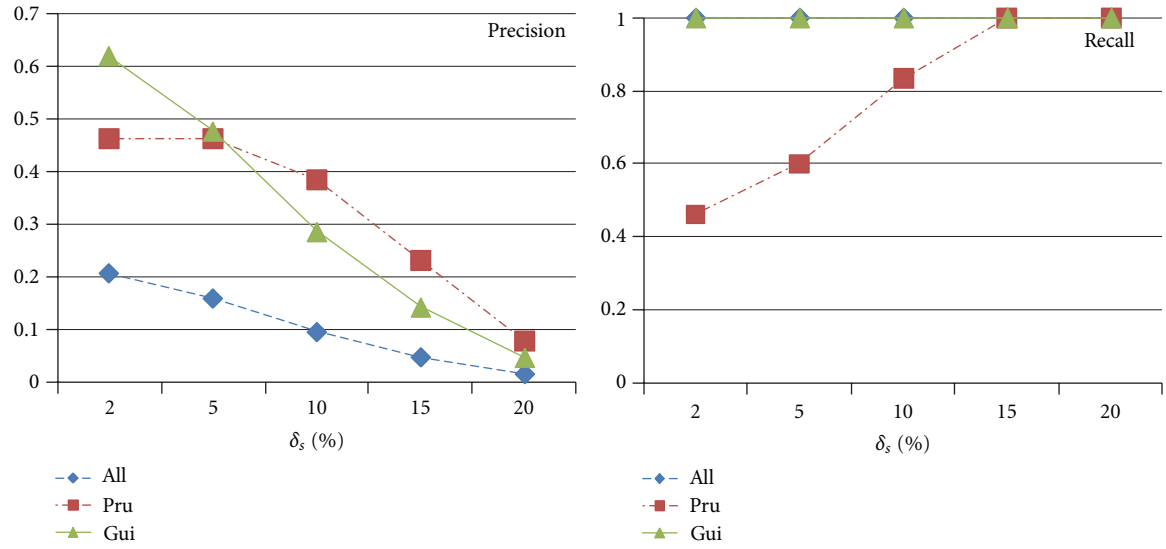


Figure 11: Effectiveness: precision and recall with respect to δ_s.

monthly macroclusters are much larger than the micro-clusters, but most of them are the trivial ones. Only 0.1% to 0.5% of those macroclusters are significant. When δ_t increases, more clusters can be merged together and the numbers of macroclusters decrease rapidly. But the numbers of significant macroclusters are more stable. Since those significant clusters have already integrated a large amount of micro-clusters, they can hardly merge with each other due to large difference on spatial and temporal features. In Figure 12(b) we record the numbers of atypical clusters with different δ_d. The influence of δ_d is smaller than δ_t. The number of significant cluster is also robust to this parameter.

We also study the influences of similarity threshold δ_{sim} and the balance function g in (5) and (6). Since they only influence the cluster integration results, we carry out the integration process with various balance functions, including max, min, the arithmetic mean (avg), the geometric mean (geo) and harmony mean (har). Figure 13 shows the average severity of the significant clusters with respect to δ_{sim}. Generally speaking, the max function integrates more clusters, and the min function is the most conservative. The differences among avg, geo, and har are minor. Hence we suggest that the users may choose a mean function (e.g., avg) as the balance function g.

From Figures 12 and 13, we can also learn that the result of significant cluster is robust to the time interval threshold δ_t and distance threshold δ_d, but the severity of significant clusters may reduce rapidly with larger similarity threshold δ_{sim}. The reason is that no atypical cluster is totally the same with another one. If δ_{sim} is too high, the micro-clusters cannot be merged and no significant clusters can be generated. In addition, δ_{sim} should be set larger than 0.5, since the clusters that merged together must be both spatially close and temporally related. Based on the experiment results, we suggest setting δ_{sim} around 0.6; in such way the aggregation clustering algorithm generates a few significant clusters with high severities.

5.4. Drill-Through Query Processing. At last we test the performances of drill-through query processing methods. In this experiment, the spatial range is no longer fixed to Los Angeles; instead it is set to a very narrow range of random road segments. In this way, the system has to drill through the basic atypical clusters to the original sensor dataset.

Since the algorithm is a two-stage process, we evaluate both stages separately in the experiments. We also compare the drill-through stage with indexes and the one without. Figure 14 shows their time costs *w.r.t.* the number of involved cells. Note that the axis of query time is in logarithmic scale. It is clear that the first stage of approximate query is much faster than the second stage of precise query. In drill-through queries, the I/O overhead is the dominant factor. The stage 1 of approximate query does not access to the detailed sensor data, so it is very efficient, and the results are returned to the users in less than ten seconds.

6. Extensions and Discussions

In this study, we illustrate the atypical cluster techniques mainly on spatial-temporal dimensions and carry out the performance evaluation on the sensor data in traffic system, because (1) the spatial and temporal dimensions are actually the most basic and important dimensions in many CPS applications, and the user's queries are usually related to such two dimensions; (2) large volume of sensor data in traffic applications is open to public; (3) the atypical events of traffic (i.e., the congestions) are actually more complex than in many other domains. Apart from spatial and temporal dimensions, the users may require to analyze

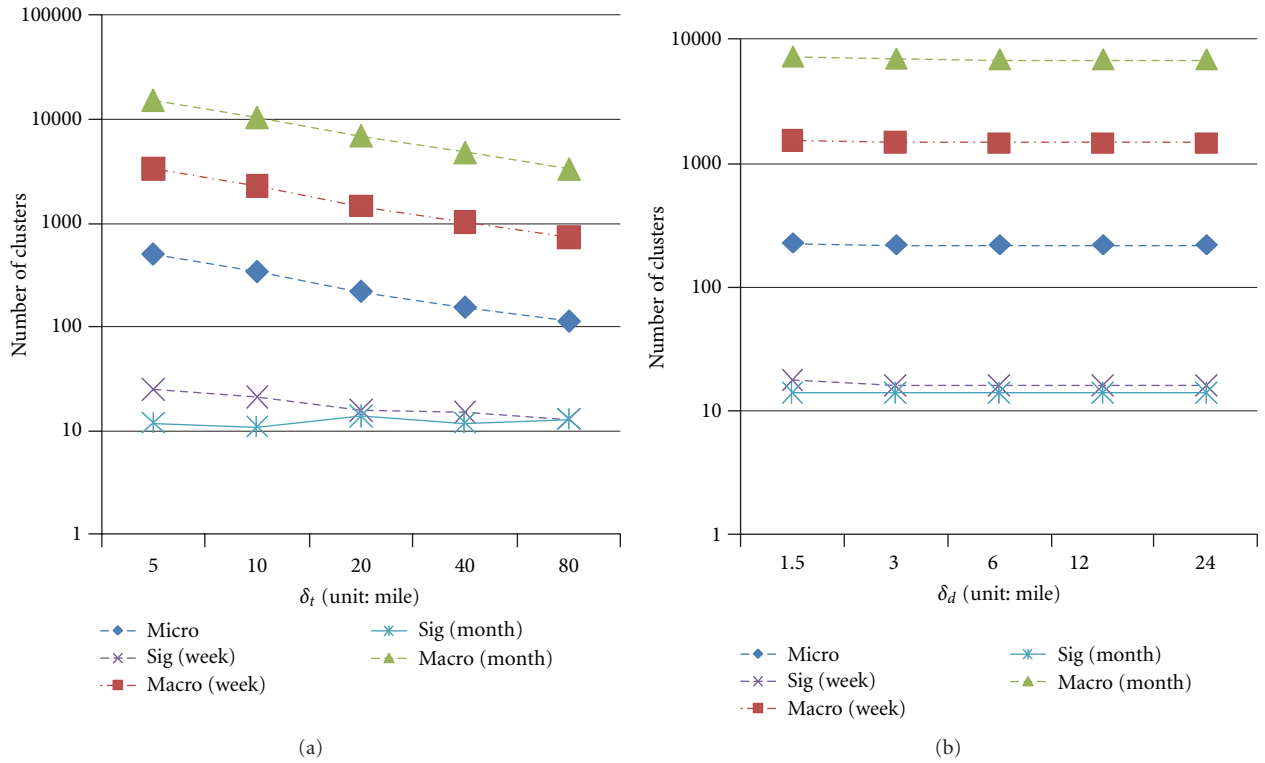


Figure 12: Size: no. of clusters versus δ_t and δ_d.

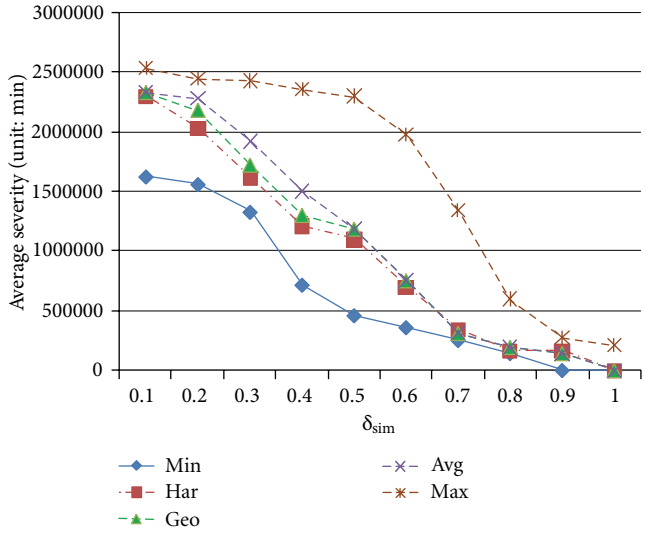


Figure 13: Average severity of significant cluster versus δ_{sim}.

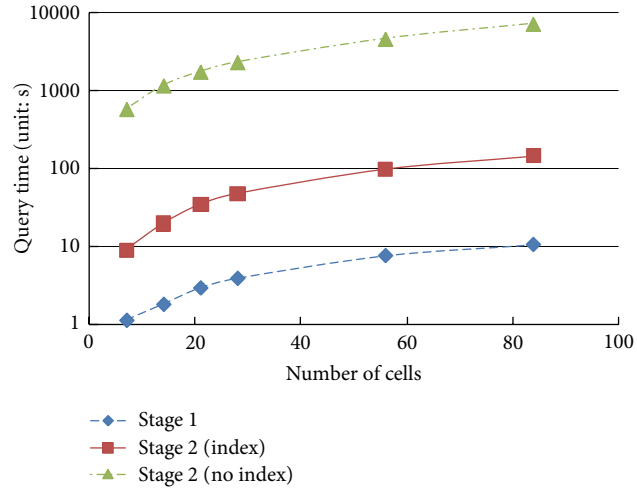


Figure 14: Efficiency: different stages of drill-through queries.

the data on other domain specific dimensions. For example, in the traffic system, the transportation officer may want to check the congestions related to bad weather or the accident reports. The proposed framework can be easily extended to support the analysis on such context dimensions. The weather dimension can be joined with temporal dimension with the date, and the accident dimension can be joined with temporal and spatial dimensions by the accident time and location. Figure 15 shows a snowflake schema of the atypical cube with dimensions of temporal, spatial, weather, accident, and so on.

By joining those dimension tables, the system can support OLAP queries on more dimensions. For instance, if users want congestion reports related to traffic accidents, the system will first select out the region W' and time window set

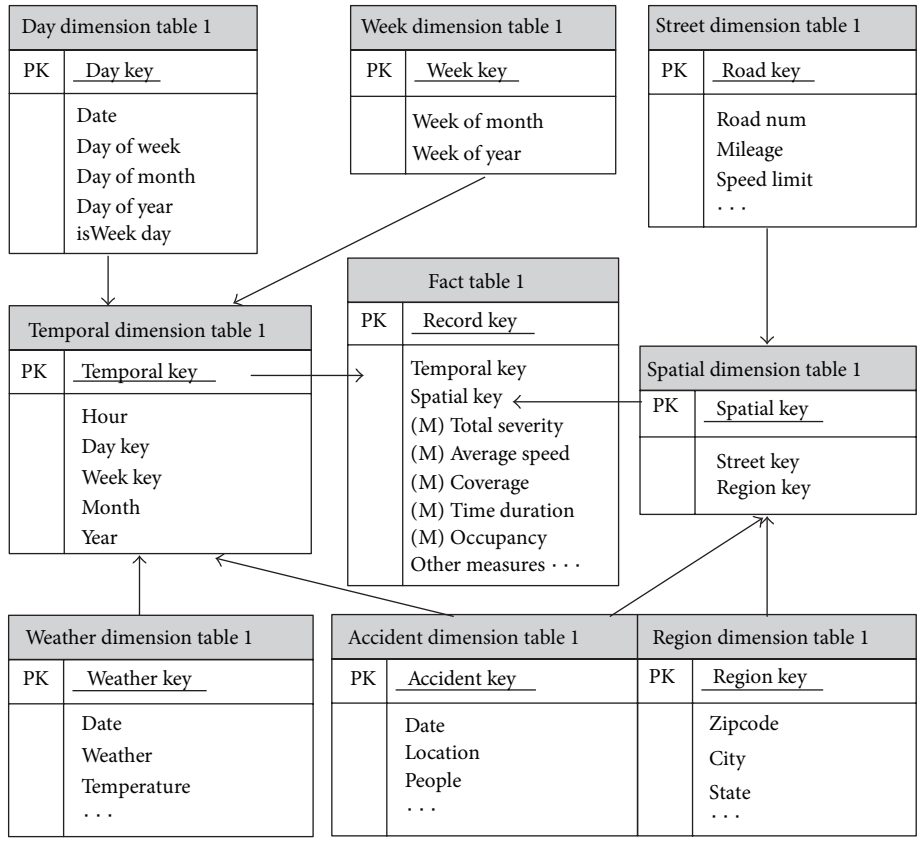


Figure 15: Atypical cube schema for congestions.

T' related to those accidents, then process query $Q(W', T')$ by the red-zone guided clustering algorithm to get the results.

In the cube construction component, the major time cost is on the preprocessing step, since the system has to scan the original datasets with huge I/O overhead. However, the preprocessing step only needs to carry out once for constructing different cubes. The massive data can also be pre-processed by the sensors themselves in a distributed manner [17]. In such a way, the amount of data and events can be reduced. Due to the hardware limitations, the sensors are likely to report some error messages and untrustworthy atypical data may be generated then. Since the main theme of this study is on multidimensional analysis of atypical data, we assume that the clean and trustworthy records can be retrieved by CPS. In our previous studies, we have proposed several methods to retrieve the atypical events from untrustworthy sensors and carry out trustworthiness analysis for the sensor networks. More details can be found in [3, 18].

The clustering and integration methods used in this study are all "hard-clustering"; that is, a micro-cluster could only be merged to one macro-cluster. Hence it is possible that the clustering result may not be deterministic. However, the influence is limited for the analytical query, because the macroclusters are usually aggregated from hundreds of micro-clusters, and there is almost no difference on merging a single micro-cluster or not.

7. Related Works

According to the methodologies, the related works of atypical cube can be loosely classified into two categories: the CPS applications and the spatial and temporal data warehousing.

7.1. CPS Applications. *PeMS* is a cyber physical system of freeway performance monitoring in California [1]. It collects gigabytes data each day to produce useful traffic information. PeMS obtains the data in the frequency from every 30 seconds to 5 minutes from each district. The data are transferred through the wide area network to which all districts are connected. PeMS uses commercial-of-the-shelf products for communication and calculation [19]. A g-factor-based algorithm [20] is used to estimate the average vehicle speed from collected data.

CarWeb is a platform to collect real-time GPS data from cars [2]. When sufficient information has been collected, the system estimates traffic information such as the average speed of vehicles. Several algorithms are employed to estimate more traffic measures.

Google Traffic is a service based on the Google Maps [21]. The feature package was officially launched in February 2007. It automatically includes real-time traffic flow conditions to the maps of thirty major cities of the United States. In a later

released version a traffic model is used to predict the future traffic situation based on historical data.

Most CPS applications do not support OLAP queries. Some of them, like Google traffic, provide prediction functions but still do not support analysis on historical data.

7.2. Spatial and Temporal Data Warehousing. The pioneering work on spatial data warehousing is proposed by Stefanovic et al. with the concepts of *spatial cube* [22]. Spatial cube is a data cube where some dimension members are spatially referenced on a map [23].

In [24], Giannotti and Pedreschi summarize the ideas of *trajectory cube*. The motivation is to transform raw trajectories to valuable information that can be utilized for decision-making purposes in ubiquitous applications. The system supports two kinds of measures [22, 25, 26]: (1) spatial measures represented by a geometry and associated with a geometric operators and (2) numerical values obtained using a topological or a metric operator.

Shekhar et al. propose a web-based visualization tool for intelligent transportation system called *Cubeview* [14]. It is aimed to investigate high-performance critical visualization techniques for exploring real-time and historical traffic data. Based on Cubeview, the *Advanced Interactive Traffic Visualization System* (*AITVS*) is implemented by using two or more distinct views to support the investigation [15]. A traffic incident detection module is also developed by considering both spatial and temporal information [27].

Papadias et al. design efficient OLAP operations based on *R*-tree index [16]. The *aggregation R-tree* defines a hierarchy among MBRs that forms a data cube lattice. In a later study [28], the authors extend the indexing techniques to spatial and temporal dimensions. *Historical RB-tree* is built to help aggregating the measures on static and dynamic regions. The *aggregate point-tree* is proposed to solve range aggregate queries [29]. In [30], Tao et al. combine sketches with spatiotemporal aggregate indexes to solve the distinct counting problem.

However, those spatial OLAP techniques are not feasible for warehousing the atypical events in CPS data. The main reason is about their measures: most methods employ COUNT, SUM, AVG and other numeric measures and aggregate them in predefined hierarchies. They cannot describe the complex atypical events. In addition, those spatial aggregations must be carried out in predefined regions (e.g., *R*-tree rectangle, zipcode area, etc.), but the atypical events may not follow their fixed boundaries. Table 5 summarizes the differences among atypical cube and some related methods.

Table 5: Comparison of related methods.

Name	Measure	Analytical query	Event integration	Fixed boundary
PeMS	Traffic speed	No	No	No
CarWeb	Traffic speed	No	No	No
Spatial Cube	Count, sum, etc.	Yes	No	Yes
CubeView	Avg speed	Yes	No	Yes
R-Tree OLAP	Count, sum, etc.	Yes	No	Yes
Atypical Cube	Atypical cluster	Yes	Yes	No

8. Conclusions and Future Work

In this paper, we have investigated the problem of multidimensional analysis of atypical sensor data in cyber-physical systems. A novel model of atypical cluster is designed to describe the atypical events in CPS data. The atypical cube is constructed as the forest of atypical clusters. The significant cluster is introduced for effective query execution, and the red-zone guided clustering algorithm is proposed to efficiently retrieve the significant clusters. Our experiments on large real datasets show the feasibility and scalability of proposed methods.

This paper is our first step in the CPS data analysis. In the future we will extend the atypical event analysis to support more complex applications, such as the event prediction and trustworthiness analysis in atypical data. We are also interested in applying the proposed methods to more applications, such as intruder detection on battlefields.

Acknowledgments

The work was supported in part by US NSF grants IIS-0905215, CNS-0931975, CCF-0905014, IIS-1017362, the US Army Research Laboratory under Cooperative Agreement no. W911NF-09-2-0053 (NS-CTA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] T. Choe, A. Skabardonis, and P. Varaiya, "Freeway performance measurement system (pems): an operational analysis tool," in *Proceedings of the 81st Annual Meeting of Transportation Research Board*, Washington, DC, USA, 2002.

[2] C. H. Lo, W. C. Peng, C. W. Chen, T. Y. Lin, and C. S. Lin, "CarWeb: a traffic data collection platform," in *Proceedings of the 9th International Conference on Mobile Data Management (MDM '08)*, pp. 221–222, Beijing, China, April 2008.

[3] L. A. Tang, X. Yu, S. Kim, J. Han, C. C. Hung, and W. C. Peng, "Tru-alarm: trustworthiness analysis of sensor networks in cyber-physical systems," in *Proceedings of the 10th IEEE International Conference on Data Mining (ICDM '10)*, pp. 1079–1084, Sydney, Australia, December 2010.

[4] L.-A. Tang, Y. Zheng, J. Yuan et al., "On discovery of traveling companions from streaming trajectories," in *Proceedings of the 28th International Conference on Data Engineering (ICDE '12)*, Washington, DC, USA, 2012.

[5] L.-A. Tang, Y. Zheng, X. Xie, J. Yuan, X. Yu, and J. Han, "Retrieving knearest neighboring trajectories by a set of point locations," in *Proceedings of the 12th International Symposium (SSTD '11)*, Advances in Spatial and Temporal Databases, pp. 223–241, Minneapolis, Minn, USA, 2011.

[6] Y. Zheng and X. Zhou, *Computing with Spatial Trajectories*, Springer, 2011.
[7] "Supplement to the presidents budget for fiscal year 2008," The Networking and Information Technology Research and Development Program, 2007.
[8] L.-A. Tang, X. Yu, S. Kim et al., "Multidimensional analysis of atypical events in cyberphysical data," in *Proceedings of the 28th International Conference on Data Engineering (ICDE)*, Washington, DC, USA, 2012.
[9] L. A. Tang, B. Cui, H. Li, G. Miao, D. Yang, and X. Zhou, "Effective variation management for pseudo periodical streams," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '07)*, pp. 257–268, Beijing, China, June 2007.
[10] X. Yu, L.-A. Tang, and J. Han, "Filtering and refinement: a two-stage approach for efficient and effective anomaly detection," in *Proceedings of the 9th IEEE International Conference on Data Mining*, Miami, Fla, USA, 2009.
[11] http://pems.dot.ca.gov/.
[12] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2nd edition, 2006.
[13] J. Gray, S. Chaudhuri, A. Bosworth et al., "Data cube: a relational aggregation operator generalizing group-by, cross-tab, and sub-totals," *Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 29–53, 1997.
[14] S. Shekhar, C. Tien Lu, R. Liu, and C. Zhou, "Cubeview: a system for traffic data visualization," in *Proceedings of the IEEE 5th International Conference on Intelligent Transportation Systems*, 2002.
[15] C. T. Lu, A. P. Boedihardjo, and J. Zheng, "AITVS: advanced interactive traffic visualization system," in *Proceedings of the 22nd International Conference on Data Engineering (ICDE '06)*, p. 167, Atlanta, Ga, USA, April 2006.
[16] D. Papadias, P. Kalnis, J. Zhang, and Y. Tao, "Efficient olap operations in spatial data warehouses," in *Proceedings of the 7th International Symposium (SSTD '01)*, Advances in Spatial and Temporal Databases, Redondo Beach, Calif, USA, 2001.
[17] X. Y. Xiao, W. C. Peng, C. C. Hung, and W. C. Lee, "Using sensorranks for in-network detection of faulty readings in wireless sensor networks," in *Proceedings of the 6th ACM International Workshop on Data Engineering for Wireless and Mobile Access (MobiDE '07)*, pp. 1–8, June 2007.
[18] L. A. Tang, Q. Gu, X. Yu et al., "Intrumine: mining intruders in untrustworthy data of cyber-physical systems," in *Proceedings of the SIAM International Conference on Data Ming (SDM '12)*, 2012.
[19] C. Chen and P. Varaiya, "An empirical assessment of traffic operations," in *Proceedings of the 16th International Symposium on Transportation and Traffic Theory*, pp. 105–124, Elsevier, 2005.
[20] Z. Jia, C. Chen, B. Coifman, and P. Varaiya, "The PeMS algorithms for accurate, real-time estimates of g-factors and speeds from single-loop detectors," in *Proceedings of the 4th IEEE Intelligent Transportation Systems Proceedings*, pp. 536–541, August 2001.
[21] http://maps.google.com/.
[22] N. Stefanovic, J. Han, and K. Koperski, "Object-based selective materialization for efficient implementation of spatial data cubes," *IEEE Transactions on Knowledge and Data Engineering*, vol. 12, no. 6, pp. 938–958, 2000.
[23] H. J. Miller and J. Han, *Geographic Data Mining and Knowledge Discovery*, CHAPMAN, 2009.
[24] F. Giannotti and D. Pedreschi, *Mobility, Data Mining and Privacy*, Springer, 2008.
[25] S. Rivest, Y. Bédard, M. J. Proulx, M. Nadeau, F. Hubert, and J. Pastor, "SOLAP technology: merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysis of data," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 60, no. 1, pp. 17–33, 2005.
[26] S. Rivest, Y. Bédard, and P. Marchand, "Toward better support for spatial decision making: defining the characteristics of spatial on-line analytical processing (SOLAP)," *Geomatica*, vol. 55, no. 4, pp. 539–555, 2001.
[27] Y. Jin, J. Dai, and C.-T. Lu, "Spatial-temporal data mining in traffic incident detection," in *Proceedings of the SIAM Data Mining Conference (SDM '06) Workshop: Spatial Data Mining*, 2006.
[28] D. Papadias, Y. Tao, P. Kalnis, and J. Zhang, "Indexing spatio-temporal data warehouses," in *Proceedings of the 18th International Conference on Data Engineering (ICDE '02)*, pp. 166–175, San Jose, Calif, USA, March 2002.
[29] Y. Tao and D. Papadias, "Range aggregate processing in spatial databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 12, pp. 1555–1570, 2004.
[30] Y. Tao, G. Kollios, J. Considine, F. Li, and D. Papadias, "Spatio-temporal aggregation using sketches," in *Proceedings of the 20th International Conference on Data Engineering (ICDE '04)*, pp. 214–225, Boston, Mass, USA, April 2004.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 489794, 21 pages
doi:10.1155/2012/489794

Review Article

Enabling Cyber Physical Systems with Wireless Sensor Networking Technologies

Chih-Yu Lin,[1] Sherali Zeadally,[2] Tzung-Shi Chen,[3] and Chih-Yung Chang[1]

[1] *Department of Computer Science and Information Engineering, Tamkang University, New Taipei City 25137, Taiwan*
[2] *Department of Computer Science and Information Technology, University of the District of Columbia, Washington, DC 20008, USA*
[3] *Department of Computer Science and Information Engineering, National University of Tainan, Tainan 700, Taiwan*

Correspondence should be addressed to Chih-Yung Chang, cychang@mail.tku.edu.tw

Received 17 February 2012; Accepted 18 March 2012

Academic Editor: Joel Rodrigues

Over the last few years, we have witnessed a growing interest in cyber physical systems (CPSs) that rely on a strong synergy between computational and physical components. CPSs are expected to have a tremendous impact on many critical sectors (such as energy, manufacturing, healthcare, transportation, and aerospace) of the economy. CPSs have the ability to transform the way human-to-human, human-to-object, and object-to-object interactions take place in the physical and virtual worlds. The increasing pervasiveness of wireless sensor networking (WSN) technologies in many applications makes them an important component of emerging CPS designs. We present some of the most important design requirements of CPS architectures. We discuss key sensor network characteristics that can be leveraged in CPS designs. In addition, we also review a few well-known CPS application domains that depend on WSNs in their design architectures and implementations. Finally, we present some of the challenges that still need to be addressed to enable seamless integration of WSN with CPS designs.

1. Introduction

Recent advances in wireless communications, networking, and embedded system technologies have led to a growing interest in developing Cyber Physical Systems (CPSs) for various purposes. In recent years, the CPS has emerged as a promising technology that can support the human-to-human, human-to-object, and object-to-object interactions in the physical and virtual worlds.

A CPS is the integration of abstract computations and physical processes [1–3], where sensors, actuators, and embedded devices are networked to sense, monitor, and control the physical world. In contrast to traditional embedded systems, the CPS is a network of interacting appliances with physical inputs and outputs instead of standalone devices. A typical CPS application is to connect appliances embedded with sensor nodes (which are responsible for information collection from the physical world as the source of CPS inputs) to some real-time decision making system (which represents the virtual world). Upon receiving the inputs from sensor nodes, the CPS will make a corresponding decision based on the inputs and computational processing to the actuators in the physical world by a sequence of control processes. We summarize below four major features of CPSs [3].

The first feature of CPS is the *integration of appliances with different communication protocols*. The appliances in the physical world might adopt different communication protocols such as WiFi, Bluetooth, Zigbee, RF, infrared and so on. The CPSs could integrate these appliances into one network. The second feature is the *rapid change of network topology*. Some wearable sensors might be worn by people. As a result, the network topology dynamically changes with the movements of people. The third feature is *remote Internet access*. Each appliance in a CPS application must have the capability to access the Internet. Based on this capability, the real-time decision making system which is part of the CPS could successfully receive available CPS inputs from appliances and then makes decisions to control the physical world. The fourth feature is the *real-time constraint* for some

delay-sensitive applications such as applications of healthcare and emergency real-time systems. If this constraint is not achieved, such delay-sensitive applications might become unreliable and unusable.

The CPS technology could efficiently manage, monitor, and indeed control the physical world. Consequently, many CPS applications are being proposed currently. These applications can be roughly classified into *smart space*, *healthcare*, *emergency real-time system*, *environmental monitoring and control* as well as *smart transportation*.

For a smart space application, many daily activities can be performed more intelligently and conveniently by the interactions between the physical world and the virtual world. A healthcare application could acquire vital signs by medical sensors worn by patients or elders. The acquired data can then be used by some real-time decision making system to determine the appropriate actions that need to be taken. An emergency real-time system could not only help people avoid unpredictable disasters (such as tsunami, volcanic eruptions or mudslide) but can also provide potential escape solutions for people. As a result, life would be safer and more secure. In the case of environmental monitoring and control applications, sensor nodes might be deployed in the outdoor environments to monitor soil moisture, air quality, and so forth. When certain specific events occur, the real-time decision making system can send commands to actuators to execute the corresponding tasks. For example, when a humidity sensor detects that the soil is too dry, the real-time decision making system will send commands to an actuator to water the dry soil. Smart transportation is one of the most important CPS applications. Sensor nodes (such as accelerometer and GPS receiver) could be embedded in vehicles to improve the traffic safety and efficiency. For instance, the accelerometer can be used to detect the potholes on the road. When a pothole is detected by an accelerometer embedded in a vehicle, the vehicle will send the location information (which is obtained by a GPS receiver) about this pothole to its nearby vehicles, hence improving the traffic safety and efficiency.

For all the aforementioned applications, Wireless Sensor Network (WSN) technology is an integral component of CPS designs. If WSN technology is not used in the development of CPSs, the real-time decision making system might have difficulty in acquiring available CPS inputs and making timely decisions. As a result, the CPS designs would be unreliable and unpredictable. It is our hope that the results of this work will help designers and researchers of CPSs to improve the reliability and predictability of such systems using sensor networking technologies.

In the next section, we present the design of a CPS architecture and the major requirements of CPSs. In Section 3, we briefly present the design of a typical sensor node architecture. We review the five fundamental WSN characteristics (such as deployment, localization, coverage, etc.) that can be leveraged in CPS designs in Section 4. In Section 5, we present a survey of well-known CPS applications from different domains and highlight their key characteristics. We discuss some of the design challenges of CPSs in Section 6. Finally, Section 7 makes some concluding remarks.

2. Cyber Physical Systems

This section presents the basic features and requirements of a typical CPS architecture.

2.1. CPS Architecture. The CPS is similar to the traditional embedded system, which aims to combine the physical processes with abstract computations. However, unlike traditional embedded systems, the CPS is a network of interacting appliances with physical inputs and outputs instead of standalone devices.

Figure 1 illustrates the architecture of a CPS which is mainly composed of *physical layer* and *virtual layer*. At the physical layer, sensors and actuators are responsible for information collection and controlling the physical world, respectively. In addition, the different types of collected information by sensors are also converted from the analog format into the digital format in this layer, and then sent to the virtual layer as the CPS inputs of the real-time decision making system. In the virtual layer, upon receipt of the inputs, the decision making system executes the abstract computations to analyze the collected data and then relays its decision to the actuators in the physical world by a sequence of control processes.

For example, as shown in Figure 2, Correll et al. [4] develops a distributed autonomous gardening system, where gardening robots and plants are networked by sensors and actuators. Robots which are all mobile actuators are capable of locating and watering plants in the garden. In addition, each plant is equipped with a humidity sensor to monitor the soil moisture. When a humidity sensor detects that the soil is too dry, it sends a request to the decision making system. Upon receiving the request, the system sends a command to a robot to water the dry soil through a sequence of control processes.

2.2. Design Requirements for CPS Architectures. Reliability and *predictability* are two important requirements in CPS designs [1–3]. This is because the quality of service (QoS) of CPS applications, such as emergency real-time system and healthcare, highly depends on these two factors. When sensor technologies are integrated with CPSs, it is a challenge for the decision making system to ensure reliability and predictability.

Deployment involves how to place sensor nodes over the given monitoring region in an efficient way while localization approach aims at providing location information for sensor nodes. Any coverage method requires that the region of interest (ROI), where the interesting events might happen, has to be covered by sensors. A data gathering scheme ensures that the collected information can be successfully delivered from sensors to the sink node (which can be treated as the real-time decision making system). Moreover, to ensure the negotiation of any two neighboring sensors and to conserve the energy consumption, the communication (Medium Access Control) support should also be considered in CPS designs.

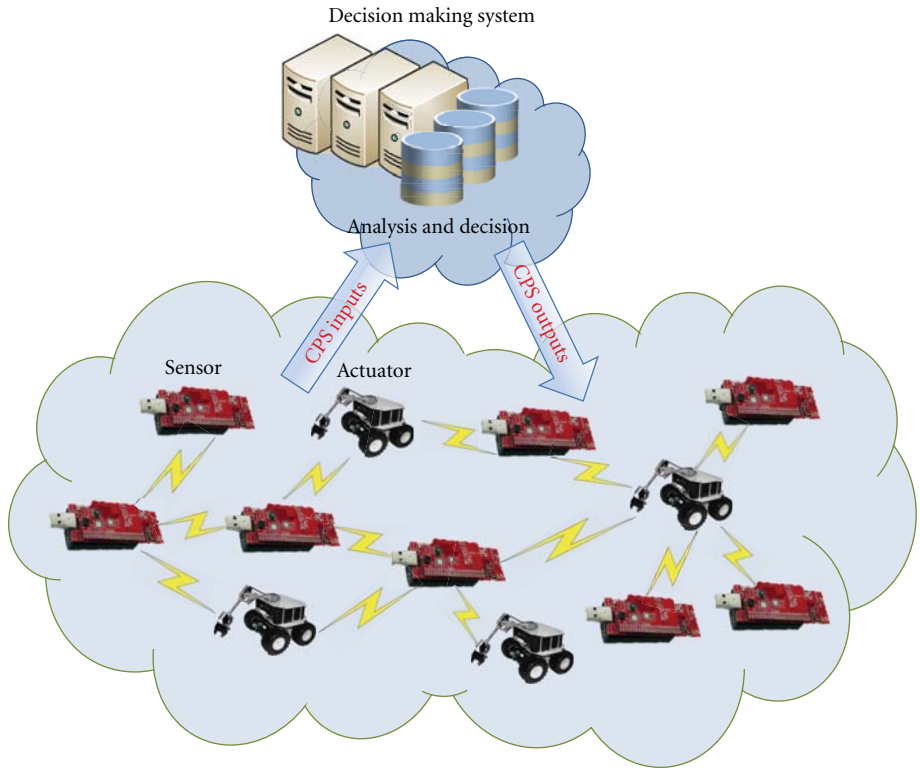


Figure 1: The CPS architecture.

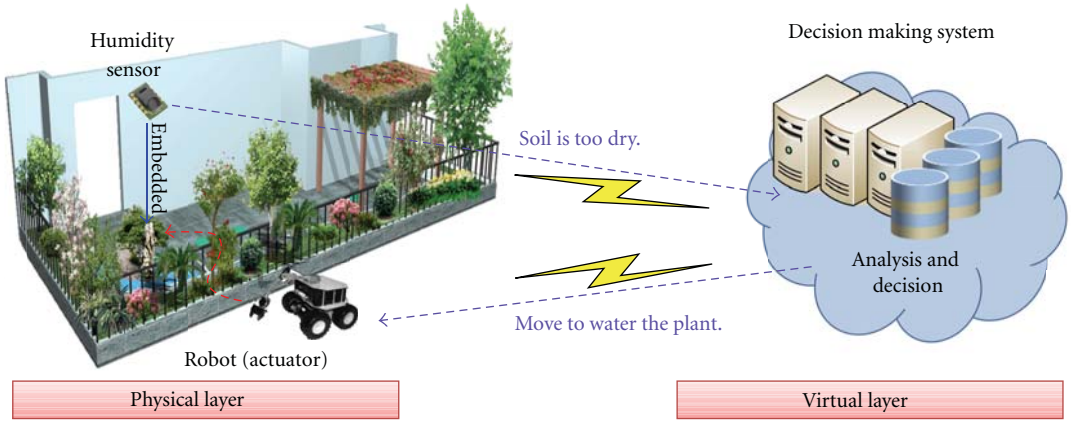


Figure 2: The distributed autonomous gardening system.

3. Wireless Sensor Node Architecture

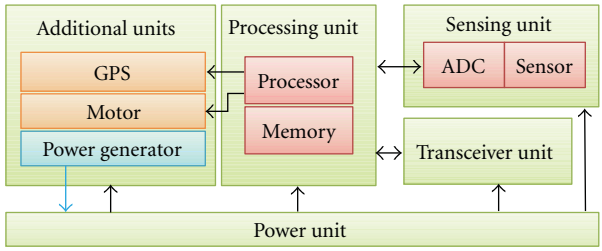
This section describes the hardware architecture of a wireless sensor node. We then introduce several popular sensor node platforms available today.

3.1. Sensor Node Architecture. The wireless sensor node is a device that converts the various measurement metrics used for physical, chemical, biomass quantities and so on in the physical world into digital information which can be read and identified by a user or by an instrument. Figure 3 illustrates the sensor node architecture, which is mainly composed of four basic components, namely *sensing unit*, *processing unit*, *transceiver unit*, and *power unit* [5].

The sensing unit consists of two subunits: the *sensor* and the *Analog to Digital Converter* (ADC). The sensor subunit is responsible for information collection from the physical world. There are many kinds of sensors currently in use in many areas of daily life. For example, temperature and humidity sensors are used to detect the temperature and humidity in the air, respectively, while light sensor can measure the intensity of the light. In addition to the sensor

Table 1: Comparison of the popular sensor node platforms.

Platform	Microcontroller	Clock (MHz)	RAM/Flash/EEPROM	Radio Transceiver	Freq. (MHz)	OS
WeC	Atmel AT90LS8535	4	512 B/8 KB/32 KB	RFM TR100	916.5	TinyOS
Rene	Atmel AT90LS8535	4	512 B/8 KB/32 KB	RFM TR100	916.5	TinyOS
Mica	Atmel Atmega 128L	4	4 KB/128 KB/512 KB	RFM TR100	916.5	TinyOS
Mica2	Atmel Atmega 128L	8	4 KB/128 KB/512 KB	Chipcon CC1000	900	TinyOS
Mica2DoT	Atmel Atmega 128L	4	4 KB/128 KB/512 KB	Chipcon CC1000	900	TinyOS
MicaZ	Atmel Atmega 128L	8	4 KB/128 KB	Chipcon CC2420	2400	TinyOS
Telos	TI MSP430	8	10 KB/48 KB/1 MB	Chipcon CC2420	2400	TinyOS
BTNode	Atmel Atmega 128L	8	4 KB/128 KB/4 KB	ZV4002 BT/CC1000	2400	TinyOS
iMote	Zeevo ZV4002	12	64 KB/512 KB	Zeevo BT	2400	TinyOS
iMote2	Intel PXA 271	13–416	32 MB/32 MB	Chipcon CC2420	2400	TinyOS



GPS: global positioning system
ADC: analog to digital converter

Figure 3: The hardware architecture of a wireless sensor node.

subunit, the ADC is used to convert the analog signals produced by sensor subunit into digital signals which are then sent to the processing unit of a sensor node.

The processing unit consists of two subunits: the *memory* and the *processor*. Similar to the storage device such as the hard disk of the host, the memory subunit is employed to store the information collected by sensing unit and is operated by the firmware. Moreover, the tasks of the processor subunit, which is similar to the central processing unit (CPU) of the host, are to execute the instructions stored in the memory subunit in addition to managing and coordinating all units.

The transceiver unit and power unit are both important components of a sensor node. Since sensor nodes might be deployed over a large-scale outdoor environment, it is difficult to use the wired transmissions to communicate with their neighbors. Moreover, replacing sensor nodes is difficult for certain applications when some nodes exhaust their energy. To cope with these two constraints, each sensor node is equipped with both transceiver and power units. The transceiver unit ensures that each sensor node can communicate with its neighbors via wireless communications while the power unit is used to manage and allocate the power resource. In general, the power source of a sensor node is usually based on batteries.

In addition to the aforementioned sensing, processing, transceiver, and power units, a sensor node might additionally have some specific components, such as the *Global Positioning System* (GPS), *motor*, and *power generator units* (as shown in Figure 3). The GPS unit can help a sensor node acquire its own location information while the motor unit offers a sensor node movement capability. The power generator unit is responsible for power generation by applying some specific technique such as solar cell.

3.2. Popular Sensor Node Platforms. This subsection presents a few popular sensor node platforms. Table 1 shows some key features of these platforms.

UC Berkeley's Smart Dust project [6] developed the Commercial Off-The-Shelf (COTS) Dust [7]. The WeC mote is one of the first platforms developed in this project. Based on the WeC mote, an important platform, namely Rene, was developed later and it is also one of the early commercialized platforms produced by Crossbow. The Rene mote later evolved into several popular platforms, including Mica, Mica2, Mica2Dot, and MicaZ.

The Mica platform has a similar performance with WeC and Rene motes in terms of radio robustness since all of them adopt the RFM TR1000 radio transceiver. The Mica platform has more memory (such as 4 KB of RAM, 128 KB of Flash, and 512 KB of EEPROM) than the WeC and Rene motes. In addition, the Mica2 platform is equipped with the Chipcon CC1000 radio transceiver instead of RFM TR1000 and therefore the radio of Mica2 is more robust than that of Mica. In addition to radio robustness, the Mica2 platform has a higher CPU clock than the Mica mote. On the other hand, the Mica2Dot platform is quite similar to the Mica2 mote. The major difference between them is that the size of Mica2Dot is smaller than that of Mica2. However, the performance of Mica2Dot is worse than that of Mica2 in terms of CPU clock. Unlike the aforementioned Berkeley's motes, the MicaZ platform is available in 2.4 GHz and adopts Chipcon CC2420 radio transceiver which results in a satisfactory radio communication. Today, the MicaZ platform is one of the most popular platforms in the world.

The Telos [8] is another famous sensor platform also developed by UC Berkeley. It is designed to minimize power consumption with increased software and hardware robustness as well as ease of use. To achieve these goals, the MSP430 produced by Texas Instruments is selected as the microcontroller of Telos platform. This is because the

MSP430 has the lowest power consumption in sleep and active modes compared to the other microcontrollers such as Atmel's AT90LS8535 and Atmega 128L. In addition, the MSP430 microcontroller also offers more memory, which ensures that the complex instructions can be successfully executed and additional hardware accelerator modules can be added. Finally, instead of integrating many different hardware modules into a sensor node, the Telos platform directly combines the programming, computation, communication, and sensing on a single device. This design makes the Telos platform easy to use.

The BTnode platform [9] is based on the Atmel Atmega 128L microcontroller and Zeevo ZV4002 Bluetooth module. Its CPU clock is able to reach 8 MHz and its memory consists of 4 KB of RAM, 128 KB of Flash, and 4 KB of EEPROM. Compared to the Berkeley's motes, the BTnode platform can effectively combine different appliances by a standardized interface and offers higher bandwidth. This is because the BTnode platform adopts the Bluetooth technique. However, it has higher power consumption and requires spending a long time on the connection setup.

The Intel Mote (iMote) [10] is mainly designed for industrial equipment monitoring. Unlike environmental monitoring, industrial monitoring aims to detect some specific measurements such as vibration and acceleration. Consequently, in addition to cost-effectiveness, sensor nodes for this application also need to have satisfactory CPU performance and radio reliability. To this end, the Zeevo ZV4002 is chosen to be the microcontroller of iMote, which adopts an ARM7TDMI core. Since the ZV4002's CPU clock can reach 12 MHz and contains 64 KB of RAM and 512 KB of Flash which provide enough performance to be used for *data compression* and *initial classification* and *analysis* for industrial monitoring applications. Moreover, the Zeevo ZV4002 also incorporates a Zeevo BT radio transceiver which supports the Bluetooth Scatternet technology. The Intel Mote 2 (iMote2) [11] is an advanced platform which is also suitable for industrial equipment monitoring applications. The iMote2 platform is based on the Intel PXA 271 which is a high performance microcontroller with CPU clock ranging between 13 MHz and 416 MHz. Furthermore, the iMote2 has 32 MB of RAM and 32 MB of Flash. In contrast to the iMote mote, the iMote2 platform uses the Chipcon CC2420 radio transceiver which adopts 802.15.4 radio technology instead of Bluetooth.

The popularity of sensor node platforms such as WeC, Rene, Mica, Mica2, Mica2Dot, MicaZ, Telos, BTnode, iMote, and iMote2 motes arises mainly because of their open source feature. In addition, an event-driven real-time operating system, called TinyOS [12], is used on these platforms because of its compactness and simplicity.

4. Key Characteristics of WSNs Used in CPS Designs

As we mentioned earlier, important sensor characteristics that need to be taken into account when we integrate sensor technologies into CPSs include deployment, localization, coverage, data collection, and communication (Medium Access Control). Figure 4 summarizes these characteristics.

4.1. Deployment. Developing a good deployment approach is necessary in CPS designs. The major objectives of deployment are to ensure the *monitoring quality* of the ROI and *network connectivity*. The monitoring quality of the ROI requires that the ROI has to be covered by sensors. The network connectivity ensures that the sensing data can be successfully delivered from each sensor to the sink node. Without an efficient deployment approach, both the monitoring quality and the network connectivity cannot be guaranteed. That is, the decision making system would not successfully receive the available CPS inputs, thereby making it harder to build reliable and predictable CPSs.

This subsection reviews existing few well-known current sensor deployment approaches which can be roughly classified into the categories of *fixed sensor* [13–19], *mobile sensor* [20–23], and *mobile robot deployments* [24–29].

4.1.1. Fixed Sensor Deployment. The fixed sensor deployment approaches can be further classified into *manual configuration* [13–15] and *random deployment schemes* [16–19]. The manual configuration approach is suitable for CPS applications which are built in an indoor or a small region environment such as smart space and healthcare applications. This is because the positions of all sensor nodes can manually be determined in accordance with the requirements of the CPS applications. References [13–15] employed a manual configuration approach to deploy fixed sensor nodes over the monitoring region. Manual configuration is one of the simplest ways to deploy sensors. However, this approach is impractical for a large-scale sensor network.

Another fixed sensor deployment approach is the random deployment scheme. Since the fixed sensors can be deployed by a helicopter, an aircraft or other vehicle, the random deployment scheme is quite suitable for a large-scale sensor network. However, to guarantee the monitoring quality and network connectivity, the number of deployed sensors has to be much larger than the number of actual required sensors. This would lead to a *redundant node problem*, resulting in many redundant sensors in the given monitoring region and leading to a significant hardware cost. To address this problem, various solutions [16–19] which make redundant sensors enter the sleeping mode to save energy were proposed for different WSN applications.

Li and Cassandras [16] consider a target coverage application in a given monitoring region in which a large number of sensors has been randomly deployed. The objective of [16] is to collect data from the given disconnected targets. Hence, there should be at least one sensor active nearby each target to ensure the monitoring quality. In addition, since all disconnected targets might be far from the sink node, a certain number of sensors also need to be active for maintaining network connectivity. In [16], any sensor in the monitoring region falls into one of the four states, including *sensing*, *relaying*, *sleeping*, and *dead states*. A sensor which

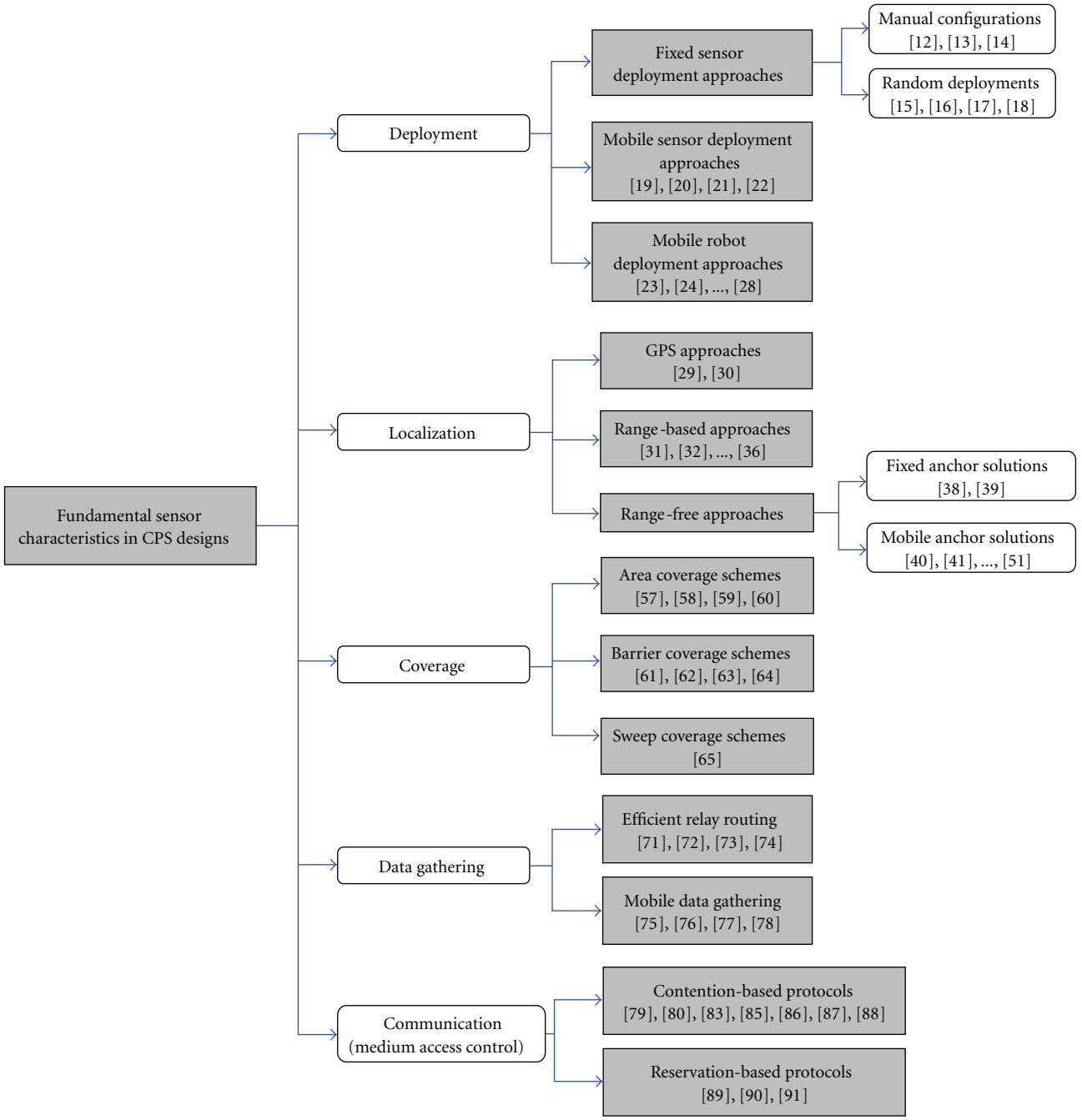


Figure 4: The five fundamental WSN technologies which are the bases in CPS designs.

falls into a sensing state should be responsible for target monitoring while a sensor in relaying state is only responsible for data relaying. When a sensor stays in the sleeping state, it does not need to participate in either monitoring or relaying tasks and enters sleeping mode to conserve its energy. A sensor falls into a dead state when it has been exhausted its energy and is no longer available to the given WSN. To address the redundant node problem and reduce the total energy consumption of sensors, the solution proposed in [16] aims at minimizing the number of sensors belonging to sensing and relaying states.

Cărbunar et al. [17] studied the problem of detecting and eliminating redundant sensors without degrading the monitoring quality in a randomly deployed WSN. The proposed solution in [17] is based on *voronoi tessellation*, which partitions the monitoring region into a number of small regions. Depending on the proposed scheme, sensors which are redundant would enter the sleeping mode, effectively prolonging the network lifetime.

Gupta et al. [18] dealt with the data gathering issue in a WSN consisting of many randomly deployed sensors. Since sensors that are close enough might contribute the same or

similar sensing data, to reduce the total energy consumption of sensors, the authors in [18] proposed one centralized and two distributed algorithms to construct a topology which consists of a subset of sensors located in the monitoring region. Only sensors involved in the constructed topology are responsible for relaying data while the other nodes enter the sleeping mode to conserve energy.

In [19], the authors also considered the data gathering issue in a randomly deployed WSN. In contrast to [18], C. Y. Chang and H. R. Chang [19] further took into account the factor of energy balancing. First, a topology construction protocol was proposed to construct a balanced data collection tree which is rooted by the sink node. Based on this protocol, the number of sensors in the left subtree and the number of sensors in the right subtree differ slight, thereby balancing the delay time for data collection. Subsequently, two node-placement techniques were proposed. Depending on the transmission loads of sensors, the two proposed schemes can be used to balance the energy consumptions of sensors. Finally, a collision-free Medium Access Control (MAC) scheduling protocol was presented to prevent collisions of packets and to further minimize the total energy consumption and delay time.

This subsection mainly introduces proposed random deployment schemes for fixed sensor deployments. Compared to the mobile sensor deployment and mobile robot deployment schemes, the random deployment approach is simpler and easier to implement. Nevertheless, the redundant node problem is still a challenge in such deployment approaches and needs to be overcome using novel solutions.

4.1.2. Mobile Sensor Deployment. The mobile sensor deployment approach is suitable for some CPS applications such as the applications of military, ecological monitoring and volcanic eruption monitoring, where the monitoring region might be too dangerous for people to reach. Therefore, using the mobility of sensor nodes to guarantee the monitoring quality and network connectivity is a good policy.

References [20–23] considered the deployment issue in a mobile WSN, where each sensor has movement capability. In the mobile sensor deployment approach, to ensure the monitoring quality and network connectivity, each mobile sensor calculates their next target location based on the information about the coverage holes. Then, it moves to the calculated target location to heal the hole. Hereafter, the coverage hole denotes the area where none of sensors' sensing ranges covers this area.

Reference [20] studied the deployment issue in a mobile WSN. The objective of [20] is to maximize the size of the area covered by mobile sensors while minimizing the movement distance of each mobile sensor. To achieve this goal, Chellappan et al. [20] translated the sensor node deployment problem into a *weighted virtual graph*. In the graph, the vertex set contains the areas in the given monitoring region while the edge set contains the possible sensor movement paths between areas. In addition, the capacities for edges model the number of sensors which can move between areas. A cost value is also assigned to each edge to capture the number of movements between areas. Based on the constructed virtual graph, the goal of maximizing the size of the area covered by mobile sensors becomes how to efficiently determine flows to the hole vertices in the graph.

Heo and Varshney [21] proposed three distributed energy-efficient deployment algorithms for mobile sensors. The first scheme operates in a peer-to-peer environment where all sensors are fairly important. Consider a mobile sensor s. The basic concept of the first scheme is to calculate the *partial forces* between sensor s and all its neighbors according to sensor s's location and the local density. Then, the *resultant force* can be derived by the calculated partial forces. As a result, sensor s can determine its movement direction based on the resultant force. The second algorithm combines the first peer-to-peer scheme with one of the current cluster-based methods. The cluster-based method, which uses a hierarchical networking concept, is employed in many WSN scenarios to take advantage of local information and to reduce energy consumption. The major difference between the first scheme and the second scheme is that each sensor can decide its own mode to be either in a clustering mode or peer-to-peer mode through its *local density* and *expected density* in the second scheme. If the local density of any sensor is close to its expected density, it changes its state to the clustering mode. Sensors which fall into the clustering mode do not need to move so as to keep the monitoring quality and conserve their limited energies. In case that the local density of a sensor is different from the expected density, the sensor changes its state to peer-to-peer mode and then executes the partial force calculation to calculate the partial forces similar to the first scheme. Furthermore, the third solution is developed based on *voronoi tessellation*. In the third solution, each sensor can estimate its lifetime in a distributed manner and then determine how long it can survive for the current network topology. Depending on the estimated lifetime, the energy efficiencies of sensors in mobility can be further increased dramatically.

Sekhar et al. [22] proposed four dynamic coverage maintenance heuristics: *Maximum Energy Based* (MEB), *MinMax Distance* (MMD), *Minimum D/E* (MDE), and *Minimum Distance Lazy* (MDL), which exploit the limited mobility of sensors to guarantee the monitoring quality and network connectivity. The major task of the four proposed approaches is to select a satisfactory hole healer when the coverage hole appears. If any sensor fails because of the limited energy or environmental causes, this leads to a coverage hole. The proposed MEB scheme chooses the neighbor of the dead node with maximum remaining energy to heal the hole. The MMD approach selects the dead node's neighbor which needs to cover the minimum distance to reach the maximum compensation for the dead node's coverage as the hole healer. The MDE method combines the objectives of the MEB and MMD schemes. It considers the ratio of maximum movement distance to the remaining energy. The neighbor with the lowest ratio of the maximum distance it can move to its remaining energy would be selected as the hole healer to heal the coverage hole. The MDL solution aims to move the hole healer with the least distance possible such that the coverage hole can be healed.

Wang et al. [23] studied the problem of placing mobile sensors to increase the quality of surveillance in WSNs. Based on the *voronoi tessellation*, two sets of distributed protocols, called *basic protocols* and *virtual movement protocols*, were proposed to control the movement of mobile sensors. The basic protocols move mobile sensors in a round-by-round manner until all sensors reach their destinations. In each round, mobile sensors initially broadcast their location to their neighbors and determine their own sensing area using the voronoi tessellation technique. If any sensor detects a hole in its responsible sensing area, it calculates an appropriate location and then moves to heal the hole. In contrast to basic protocols that move mobile sensors in a round-by-round manner, the virtual movement protocols aim at directly moving mobile sensors to their destinations instead of step by step, thereby minimizing the movement distance of each node.

This subsection has reviewed several existing mobile sensor deployment schemes [20–23]. Compared to the fixed sensor deployment solution, the mobile sensor deployment approach is suitable for a large-scale sensor network and could alleviate the redundant node problem. Furthermore, the mobile sensor deployment approach is able to deploy fewer sensors to guarantee both monitoring quality and network connectivity. Nonetheless, there are two major weaknesses in the mobile sensor deployment approach. The first is that each mobile sensor needs to incur additional hardware cost to support its mobility. The other weakness is that considerable energy consumption is required for each mobile sensor so as to move from one location to another location.

4.1.3. Mobile Robot Deployment. The mobile robot deployment approach is similar to the mobile sensor deployment solution. This mobile robot approach is also suitable for CPS applications where the monitoring region is dangerous and unreachable. However, the mobile robot deployment approach is easier to implement than the mobile sensor deployment scheme. This is because the robot with fixed sensors could efficiently deploy the fixed sensors over the monitoring region if it follows a well-designed deployment algorithm.

References [24–29] adopted the mobile robot deployment approach to deploy the fixed sensors in a given monitoring region. During the deployment process, to ensure monitoring quality and network connectivity, the robot explores the environment and deploys a fixed sensor at the target location from time to time.

In the mobile robot deployment scheme, it is a challenge to eliminate the negative impact of unpredicted obstacles. Obstacles such as walls, buildings, blockhouses and so on, may exist in the outdoor environment, dramatically influencing the performance in terms of robot deployment. A robot-deployment scheme that does not take into consideration obstacles might result in problems of coverage hole and coverage redundancy.

Batalin and Sukhatme [24] assumed that the robot is equipped with a compass which makes the robot to be aware of its movement direction. In study [24], a robot movement strategy which uses the deployed sensors to guide the robot's movement and sensor deployment, was proposed. Although the proposed robot-deployment scheme could guarantee the monitoring quality and network connectivity in an obstacle-free environment, however, it does not take into account the obstacles in the given monitoring region. The next movement of the robot is guided by the nearest sensor only. As a result, problems of coverage hole and coverage redundancy might occur when the robot encounters obstacles. Furthermore, during the robot deployment process, all deployed sensors stay in active mode to participate in the guiding tasks, leading to an energy-inefficient WSN.

The efforts described in [25, 26] aim to eliminate the negative impact of unpredicted obstacles during the robot-deployment process. Batalin and Sukhatme [25] employed the robot to deploy the fixed sensors based on the predefined direction priorities, including *north, south, west,* and *east directions*. Each sensor keeps track of the time interval that the robot does not explore for each direction. Based on the time interval, the deployed sensors within the communication range of the robot could guide the robot's movement by suggesting an adequate direction. Upon receiving suggestions from different sensors, the robot integrates these suggestions and selects the best direction for patrol and/or sensor deployment. On the other hand, the study [26] proposed another robot-deployment scheme, which includes four traveling orders, namely *random*, *cross*, *line*, and *circle*, as the movement options of the robot. Nonetheless, since each subsequent movement is determined by predefined rules regardless of the obstacles relative to the robot, both approaches proposed in [25, 26] cannot guarantee the monitoring quality and might even cause coverage redundancy when the robot encounters the obstacles. Moreover, there is no discussion about how to deal with irregular obstacles.

To address the problem that arises in [25, 26], Wang et al. [27] proposed a centralized approach which employs global obstacle information to calculate the best deployment location of each sensor. Although the proposed mechanism ensures the monitoring quality and network connectivity using fewer fixed sensors, the global obstacle information is still required. Furthermore, since the global information about obstacle is difficult to acquire in an unexplored area, the developed mechanism could only be used in some limited applications.

Studies [28, 29] developed robot-deployment algorithms that overcome unpredicted obstacles. By applying the proposed schemes in [28, 29], the robot rapidly deploys a near-minimal number of sensors to guarantee the monitoring quality and network connectivity. In [28], the proposed approach consists of a *node placement policy* and a *spiral movement policy*, where the node placement policy aims to deploy fewer sensors to achieve full coverage while the spiral movement policy is adopted as a strategy for the robot movement. On the other hand, the proposed scheme in [29] involves the designs of a node placement policy, a snake-like movement policy, and various obstacle-handling rules. The node placement algorithm minimizes the coverage redundancy of neighboring sensors while a

snake-like movement pattern is employed by the robot to deploy sensors. In addition, several obstacle-handling rules were proposed to alleviate the negative impact of unpredicted obstacles.

This subsection surveys several mobile robot deployment approaches [24–29]. Similar to the mobile sensor deployment scheme, such approaches are suitable for a large-scale sensor network and also have no redundant node problem. Indeed, after the deployment process, the robot might further execute other missions such as hole detection, redeployment, monitoring, and so forth. As a result, from the hardware cost point of view, the mobile robot deployment approach is better than the mobile sensor deployment solution.

4.2. Localization. In most CPS applications, the location information is important for the real-time decision making system. This is because every decision made by the decision making system is based on the location information of sensor nodes. The actions made by the decision making system are generally relayed to the locations where the events have occurred. We review below various recently proposed localization schemes in WSNs.

Localization with low cost and high accuracy is of utmost importance for most applications in WSNs, such as location-aware routing, target tracking, coverage and others. Without the availability of location information, these applications cannot be executed successfully. Equipping each sensor node with a GPS device [30, 31] is one of the simplest ways to help the node acquire its own location information. Based on the NAVSTAR satellite constellation, a sensor node is able to obtain its coordinates if it is located in the satellite coverage and no obstacle exists in the path of satellite signals. However, having a GPS device for each sensor node is not a feasible solution. To remove the GPS requirement from each sensor node, there are various localization approaches which have been proposed in literature. These approaches can be classified into two categories, namely *range-based scheme* [32–38] and *range-free scheme* [39–53].

4.2.1. Range-Based Localization Scheme. The range-based approach helps each sensor node acquire its own location information by using either *Euclidean distance* or *relative angle* between any two neighboring sensor nodes. The distance or angle information could be measured by the Received Signal Strength Indicator (RSSI), Angle of Arrival (AoA), Time of Arrival (ToA), or Time Difference of Arrival (TDoA) techniques [32–38]. The RSSI scheme measures the power strength at the receiver. When the sender transmits a signal, the receiver is able to estimate the distance between them according to the propagation loss. The ToA and TDoA [32] are both time-based methods, where ToA is based on the distance estimations by the signal arrival time while TDoA depends on the time difference between two consecutive arrived signals. The AoA approach [34] uses special antenna configurations to estimate the angle of arrival of the received signal from the sender. Consider a node s and three landmarks a, b, and c which have location information. Depending on the AoA approach, node s can acquire its coordinates through the means of triangulation which are based on the positions of landmarks a, b, and c and the angles $\angle asb$, $\angle asc$, and $\angle bsc$.

Research efforts described in [35, 36] employed Received Signal Strength (RSS) measurements to help nodes get their coordinates. Reference [35] developed a fine-grained indoor location sensing system based on Radio-Frequency (RF) signal strength. In [35], a central server uses the measurement of signal strength provided by each base station to estimate the distance between each base station and each node. By applying the triangulation technique, the coordinates of each node can be calculated. Study [36] considered a target tracking application in an indoor environment and also developed a RF based system. This system is based on empirical signal strength measurements and a simple signal propagation model. Similar to [35], the location of each target can be determined by the signal strength information from base stations. Although using RSS measurements [35, 36] is able to help nodes derive their location information, it might suffer from mistakes due to the random nature of the fading channel. To solve this problem, the work described in [37] employed proximity measurements that complement the existing localization approach using RSS measurements.

References [33, 38] described two time-based localization methods. Savvides et al. [38] used a minimal number of beacons, which are location-aware, to help sensor nodes learn their own location information. The proposed distributed technique, called AHLoS (Ad-Hoc Localization System), mainly consists of a *ranging phase* and an *estimation phase*. In the ranging phase, each sensor node measures the distances between itself and all its neighbors using the ToA technique. During the estimation phase, in addition to the distance information, each sensor determines its own coordinates by its neighboring beacons. The Cricket location support system [33] is based on the TDoA technique for indoor localization. Instead of only relying only on RF signals, it uses the beacons along with the combined RF and ultrasound signals to help nodes learn their coordinates.

This subsection reviews several well-known range-based localization schemes [32–38], which take advantage of the RSSI, AoA, ToA, or TDoA techniques to help sensors learn their own location information. Nonetheless, there are two drawbacks with range-based approaches. First, the nodes have to be equipped with expensive hardware, increasing their hardware costs. Second, the signal might not be received successfully because of fading channel, interference, collision and so on, greatly degrading the localization performance [54]. To deal with these two problems, various range-free localization schemes have been proposed.

4.2.2. Range-Free Localization Scheme. To minimize hardware costs for each sensor node, many range-free localization schemes [39–53] have been proposed for a resource-constrained WSN.

Unlike the range-based approach, some studies [39, 40] enable sensors to learn their own location information based on deploying several fixed anchors, which have location

information by equipping them with GPS devices or by some other means. Niculescu and Nath [39] proposed a distributed range-free localization scheme. In addition to sensor nodes, some fixed anchors are randomly and uniformly deployed in the given monitoring region. Using the help of the fixed anchors, sensors are able to calculate their own location information in a distributed manner. To start with, all fixed anchors communicate with each other to acquire the hop counts between them. According to the Euclidean distance between any pair of fixed anchors, each anchor estimates the average distance of one hop and then broadcasts the estimated distance and its own coordinates to all sensors in the monitoring region. Upon receiving different messages from at least three fixed anchors, each sensor can therefore calculate its own location information. However, in a random deployed WSN, sensors might not be deployed uniformly, resulting in degradation of the localization performance.

He et al. [40] also used a few fixed anchors to help sensors obtain their own location information. Using beacons from these fixed anchors, each sensor can determine the area in which it is located. In [40], each sensor initially selects three fixed anchors whose beacons can be received by it and checks if it is located in the triangular region formed by connecting these three anchors. This operation will repeatedly be executed until all combinations of the different audible anchors are exhausted or the required location accuracy is achieved. Afterward, each sensor node calculates the intersection of all triangular regions and then treats the center of gravity of the intersection region as its coordinates.

Although deploying a few fixed anchors does help each sensor obtain its coordinates and can reduce the hardware cost compared to range-based localization approach, however, the fixed anchors are still equipped with specific equipment such as a GPS device. To further remove the requirement of deploying a number of fixed anchors in the given monitoring region, other proposed schemes [41–53] employed a mobile anchor instead of a fixed anchor.

Studies [41, 42] exploited several mobile anchors to help sensors acquire their coordinates, where each mobile anchor has location information. Ssu et al. [41] assumed that the communication ranges of mobile anchors and sensors are identical and the shapes of them are all modeled as perfect disks. In [41], each mobile anchor randomly determines its movement direction and continuously broadcasts beacon messages including its current coordinates. When any mobile anchor enters and then leaves the communication range of node s, the first and the last coordinates received from the mobile anchor will be viewed as the coordinates of two points, which fall on the boundary of the communication disk (communication range) of node s. Both points are also the end points of a chord of sensor s's communication disk. Similarly, another chord can be obtained by node s after any mobile anchor passes through the communication disk of sensor s again. After obtaining two chords, node s calculates two perpendicular bisectors of the two chords. Since each perpendicular bisector of a chord must pass through the center point of the circle, node s can therefore determine its location which is the intersection of the two perpendicular bisectors. The basic concept of [42] is similar to that of [41]. The major difference between them is that study [42] employed aerial anchor nodes to execute the localization process. Nonetheless, the localization performances of [41, 42] depend on the frequency of beacon broadcasting. The anchors broadcasting beacon messages more frequently would result in better localization performance and higher energy consumption.

Studies [43–51] employed *the area-based localization approach* to help sensors get their coordinates. In studies [43–48], a mobile anchor being aware of its own location information moves in the monitoring region and periodically broadcasts a beacon with its current coordinates to improve the location accuracy of the nearby sensors. Upon receiving the beacon message, the sensor node indicates that it is within the region of the circle centered at the coordinates of the mobile anchor with a radius r, where r is the communication range of the mobile anchor. Therefore, the sensor node identifies that its location is within the circle region which is referred to as *estimation region*. Based on the range-constraint of beacon messages, a static sensor that receives several different coordinates from the mobile anchor might reduce its estimation region by calculating the intersection region of these estimation regions, leading to an improvement in the location inaccuracy. However, the range-constraint localization is mainly applied by those sensors that are actually one-hop neighbors of the mobile anchor.

Study [49] extends the range-constraint from one-hop to the two-hop neighboring sensors. Let $A = \{a_1, a_2, \ldots, a_n\}$ denote the set of n neighbors of the mobile anchor and B_i denote the set of sensors which are the neighbors of sensor a_i in set A. Upon receiving the location information from the mobile anchor, all sensors in set A evaluate their estimation regions and then broadcast the regions to their neighbors. Since the two-hop neighboring sensors of mobile anchor cannot receive the location information from mobile anchor, nodes in set B_i have a location constraint that they are not located in the estimative region of a_i. Nonetheless, since nodes in set B_i are all neighbors of node a_i, they have another constraint that they are located in the communication range of any possible location of a_i. Based on these two constraints, sensors in set B_i can also derive their own estimation regions.

In addition, in [43–49], the intersection of estimation regions is difficult to calculate because it is an irregular region. Other studies [50, 51] have proposed localization schemes by applying a rectangular region instead of a circular region to simplify both the calculation and the representation of the new estimation region.

Although the existing area-based localization approaches [43–51] can efficiently make each sensor obtain their estimation regions, however, these studies do not consider how the mobile anchor moves and where the beacon should be broadcasted in the given monitoring region. Furthermore, they also cannot distinguish between the relative locations of any pair of neighboring sensors. As a result, when sensors execute some location-aware applications such as routing, a poor performance might be obtained. To address these two problems, the authors of [52, 53] proposed a few techniques.

Chang et al. [52] proposed an anchor-guiding mechanism to further improve the localization performance of [43–51]. The proposed mechanism aims to determine the beacon locations and construct an efficient path for the mobile anchor passing through all beacon locations to improve the accuracy of the localization task. First, the monitoring region is partitioned into a number of grids and each grid is assigned a weight value which represents the localization benefit. Then, according to the weight value of each grid, the promising grids for broadcasting beacons are selected by the mobile anchor. Finally, a path construction algorithm is presented to construct a path passing through the selected beacon locations while minimizing the movement of mobile anchor.

Chang et al. [53] extended the existing area-based localization approach. The proposed scheme not only provides each sensor with an estimation region but also helps each pair of neighboring sensors distinguish their relative locations. The key idea of this article is to use a mobile anchor broadcasting tone signal to identify the relative locations. The proposed mechanism mainly consists of two strategies, namely *distinguishing relative location* and *path planning*. Initially, depending on the order of entering and leaving the tone-signal range, a set of rules were developed for each sensor to distinguish relative locations with all its neighbors. Then, two efficient path planning strategies were proposed for the mobile anchor to explore the whole monitoring region with low energy consumption.

This subsection has reviewed various well-known range-free localization schemes [39–53]. Although the localization performance of range-free approach is not better than that of range-based approach, however, the hardware cost of the range-free approach is much lower compared to range-based approach.

4.3. Coverage. The sensor coverage problem relates to whether we have a fixed deployment or a non-fixed deployment which are based on if deployment of sensors is to be planed before an event or after. Fixed sensor deployment implements a plan before an event occurs. It is usually based on some geographic shapes and some mathematical computation that is used to determine the position of each sensor. The geometric shapes can be a hexagon [55] or a square [56]. When sensing in a regional environment, we can prepare a priori to reach the point of interest which is closest as much as possible using regular methods such as row-by-row and column-by-column, grid, to deploy sensor nodes for fixed deployment scenarios. The main benefit of the aforementioned approach is that it can ensure that there is no hole in the coverage area of interest. The disadvantage is that the approach can be easily affected by terrain or other obstacle which increases the difficulty of the placement of sensors. In the case of the non-fixed deployment approach, we use the node that can automatically move with some technical adjustment to monitor the target location. The advantage of this approach is that it is relatively unaffected by terrain and is a good method for computing the target coverage area. The disadvantage is that the sensing range of a sensor node always overlaps with the sensing range of other nodes. To address this overlapping problem, we need more sensor nodes. However, this would increase the deployment cost. The following section describes three main classes of non-fixed sensor deployment strategies, namely full coverage [57–60], barrier coverage [61–64], and sweep coverage [65].

Full coverage includes the whole area we must cover with the sensing range of the sensors which is important for military applications. To protect a military base, we need to deploy appropriate sensor nodes to monitor the surrounding environment. The most common approach is the use of Voronoi Diagram. A group of points in the environment use the vertical line between two nearest points to study its quality of coverage.

Barrier coverage involves placing the sensor node at the center of the circle. When an object wants to pass through the area which was surrounded, it will be detected by the sensor node. In [64], the authors use attraction and repulsion to let each node find the distance between itself and neighbors and to set up a barrier to monitor the surrounding environment.

In the case of the Sweep coverage approach, the area we want to monitor has a very important Point of Interest (POI). We use a node which has the ability to move to patrol and monitor the area of interest. But when we use a node to patrol many POIs, it will result in the Traveling Salesman Problem (TSP). To address this problem, a centralized sweep algorithm called CSWEEP segmentation method [65] was proposed where we allow each segment to have a mobile sensor node to perform regular patrols. The moving route of each mobile sensor is predetermined to guarantee the coverage. But CSWEEP needs to know the POI location. For scalability, a distributed sweep algorithm named DSWEEP was proposed which enables sensors to cooperate efficiently to provide required coverage. Each sensor node decides its moving path individually at runtime using the knowledge of the traces of other sensor nodes.

In some areas covered by the sensors, they must also return data. But if using multi-hop manner to send data, it will consume too much energy. Some previous studies also deployed a sink in the monitoring region to collect data where is the sink located. Sinks can be classified into fixed [66] and variable [67] ones. While a sink is stationary in a certain position, the sensory data can be routed to the sink in an efficient way. When a sink is mobile, routing to the mobile sink on a predefined trajectory should be considered.

4.4. Data Gathering. Data gathering in WSN is defined as the systematic collection of sensed data from multiple sensors to be eventually transmitted to the base station for processing [68, 69]. The main constraint is that most sensor nodes are powered by limited battery. Thus, it becomes an important issue in data gathering to reduce the energy consumption in order to prolong network lifetime. Recent research efforts about efficient data gathering schemes can be generally classified into two categories namely, efficient relay routing and mobile data gathering.

In the case of efficient relay routing, sensed data from the environment is forwarded to the data sink via multihop relays among sensors. Data gathering techniques with aggregation have been proposed by the following researchers. Liang and Liu [70] presented a generic cost model of energy consumption for data gathering in sensor networks and proposed heuristic algorithms to solve it. Wang et al. [71] studied the data aggregation of Divisible Perfectly Compressible (DPC) functions for random WSNs. The authors designed two protocols, called Single-Hop-Length (SHL) and Multiple-Hop-Length (MHL) schemes, to derive the optimal aggregation throughput depending on a given gathering efficiency. Incel et al. [72] studied fast convergecasting in WSNs where nodes communicate using a Time Division Multiple Access (TDMA) protocol to minimize the schedule length.

With hierarchical infrastructure, Zhang et al. [68] studied two-layered heterogeneous sensor networks, where the network is partitioned into clusters and a powerful cluster head controls all sensors in a cluster. They focused mainly on the energy-efficient designs within a cluster to prolong the network lifetime.

When sensed data is highly correlated, most research efforts use source coding strategies to find an optimized rate allocation at the sensor nodes. Arjmandi and Lahouti [73] considered efficient data gathering in a WSN cluster whose cluster head is of limited complexity (memory and computational complexity) and employed asymmetric Slepian-Wolf codes. Tan et al. [74] presented a distributed resource allocation framework to maximize the network utility and proposed a dynamic network coding strategy that allows an intermediate sensor node to independently decide whether to combine incoming data flows.

In the case of mobile data gathering, the mobile data collector can move around the sensing field and collects data from the source nodes through short-range communications. The main advantage with this approach is that the mobile collector can reduce the energy consumption of routing all the sensed data to the data sink. With uncontrollable mobility, Jain et al. [75] presented an analytical model to understand the key performance metrics such as data transfer, latency to the destination, and power.

With controlled mobility, an efficient moving tour can be planned for specific purposes. Zhao et al. [76] considered the tradeoff between concurrent data uploading time and moving tour. Xing et al. [77] proposed a rendezvous scheme to combine the advantages of controlled mobility and local data caching and jointly optimizes data routing paths and the tour of the mobile collector. Fei et al. [78] formulated the moving process of data collectors as a Markov chain and determined the moving path using a Markov decision process. The collectors move along the path defined by the optimized policy which is computed off-line and downloaded to collectors in real-time.

4.5. Communication (Medium Access Control Protocols). The design of Medium Access Control (MAC) protocol plays an important role in the design of CPSs. Many CPS applications, such as the applications of military, environmental monitoring, and target tracking, are applied to the outdoor environment. Therefore, sensor nodes are difficult to be recharged when they exhaust their limited energy. This section presents various MAC protocols that have been proposed to efficiently manage the sensor nodes' energy.

Energy-efficient MAC protocols for WSNs need to conserve the energy consumption during sensor node communications. There are several attributes that should be considered in designing an efficient MAC protocol [79]. The first attribute is *energy efficiency*. Since sensors are battery powered and are often difficult to be changed or recharged, the reduction of energy consumption of each sensor node is a challenge. The second requirement is *latency*. In WSNs, the sensing data should be delivered from sensors to the sink node in a real-time manner so that the corresponding operation could be executed rapidly. The third requirement is *fairness* which ensures that all sensors are able to send their sensing data to the sink node fairly, thereby avoiding the *starvation problem*. Furthermore, in order to increase the network throughput, the *bandwidth utilization* should be also considered because of the limited bandwidth resource.

In the literature, many energy-efficient MAC protocols have been proposed for WSNs. These protocols can be broadly grouped into *contention-based* MAC protocols [79–85] and *reservation-based* MAC protocols [86–88]. We present below recently proposed contention-based MAC protocols as well as several popular reservation-based MAC protocols.

4.5.1. Contention-Based MAC Protocols. The IEEE 802.11 [80] defined a contention-based MAC protocol which was inspired by MACAW [89]. It adopts technologies including CSMA (Carrier Sense Multiple Access), CSMA/CA (CSMA/Collisions Avoidance), and Random Backoff to avoid transmission collisions and maintains fairness among wireless devices in a single-channel environment. Let nodes S and R be the sender and receiver, respectively. In the CSMA technology, the sender S has to initially listen to the channel for a predetermined period of time so as to check if there are any activities on the channel. In case that the channel is sensed "idle", sender S is allowed to access channel for transmitting data to receiver R. Otherwise, sender S has to defer its transmission. Although CSMA technology can efficiently prevent the current transmission from collision, it cannot cope with the *hidden node problem*. For example, given three nodes S, R, and H, if H is "hidden" from S, it could happen that the data signal sent from S to R cannot be sensed by node H. As a result, node H might transmit data to its receiver and hence a collision might occur at node R. To deal with the hidden node problem, the CSMA/CA technology should be applied. A sender S intending to exchange data with a receiver R should firstly send the RTS (Request to Send) packet to R. Upon receiving the RTS packet, the receiver R simply replies a CTS (Clear to Send) packet to sender S. All other nodes that receive the RTS or CTS packets should defer their transmissions until

the data exchange between S and R is completed. In addition, a Random Backoff mechanism was proposed in IEEE 802.11 MAC to prevent collisions among the transmissions of multiple RTSs. Although IEEE 802.11 MAC is widely used because of its simplicity and robustness to the hidden node problem, the energy consumption is very high when nodes stay in the idle state [90].

PAMAS [81] is one of the earliest contention-based MAC protocols and is based on MACA [91]. The difference between PAMAS and MACA is that PAMAS employs two independent radio channels to exchange control and data packets, where one channel is used to exchange RTS/CTS message and the other channel is used for data transmissions. A node which is not involved with the transmission might switch off its radio to save its energy. However, the PAMAS approach needs to use two radios in the different frequency bands, hence increasing the hardware cost and the complexity of sensor node design.

Ye et al. [79] proposed a contention-based MAC protocol, called S-MAC, which is modification of the IEEE 802.11 MAC protocol. The objective of S-MAC is to reduce the energy wastage resulting from collisions, overhearing, control packet overheads, and idle listening. In S-MAC, the time is divided into a number of equal-length frames and each frame is composed of a *listen window* and a *sleep window*. In the listen window, each node wakes up and listens whether or not any other node intends to communicate with it. If it is the case, the sender and the receiver exchange the control packets (such as SYNC, RTS, and CTS) and then exchange the data in the next sleep window. Otherwise, each node listens until the current listen window ends and then changes its state from "listen" to "sleep" and turns off its radio to conserve energy. However, the S-MAC also has high energy consumption if nodes are in the idle mode.

Van Dam and Langendoen [82] presented another contention-based MAC protocol, called T-MAC, which is similar to S-MAC. In T-MAC, the time is also divided into a number of equal-length frames and each frame is composed of a listen window and a sleep window. The major difference between T-MAC and S-MAC is that T-MAC enables the length of each listen window to be calculated dynamically. Each node, say s, wakes up at the start of each listen window and listens whether or not any other node intends to communicate with it. If no other nodes intend to communicate with node s within a predefined time interval, node s terminates its listen window and enters the sleep window.

B-MAC [83] is a contention-based MAC protocol designed for WSNs and adopts CSMA technology. To achieve low power operation, B-MAC uses an adaptive preamble sampling scheme to reduce duty cycle and minimize idle listening. In B-MAC, when no activation events occur, each node continuously sleeps for a fixed period of time t_{sleep} and then wakes up and listens if any other node intends to communicate with it. Consider a sender S intending to send data to a receiver R. Sender S initially checks if the channel is clear using Clear Channel Assessment (CCA) approach. If it is the case, sender S broadcasts preambles for a period of time t_{preamble}. To ensure that receiver R is able to receive the preamble from sender S, the relation $t_{\text{preamble}} > t_{\text{sleep}}$ should be satisfied. As a result, receiver R can be aware that sender S intends to communicate with it and then successfully receives the data from S after time interval t_{preamble}. However, heavy traffic load will worsen the performance of B-MAC. This is because the transmission delay of B-MAC increases due to the long preamble.

The WiseMAC [84] protocol was developed for WSNs. Similar to the study described in [85], the WiseMAC also adopts spatial TDMA (Time Division Multiple Access) and CSMA with preamble sampling scheme. However, in [85], all sensor nodes have to work in a multi-channel environment. In contrast, WiseMAC works in single-channel environment and uses non-persistent CSMA with preamble sampling technique to conserve energy during idle listening. In WiseMAC, when no activation events occur, each node continuously sleeps for a fixed period of time and then wakes up and listens if any other node intends to communicate with it. To cope with the long preamble problem in B-MAC, each sensor node in WiseMAC maintains all its neighbors' wakeup schedules for the data transmission. If any sender S intends to send data to receiver R, sender S initially checks the next time when receiver R will wake up. Then, sender S broadcasts preambles for a shorter period of time when receiver R wakes up.

This subsection has described some of the well-known contention-based MAC protocols for WSNs [79–85]. Compared to reservation-based MAC protocols, the contention-based schemes are simpler to implement. This is because the contention-based schemes only need local time synchronization instead of global time synchronization. In addition, they do not need to have the knowledge of network topology, reducing the communication overheads. However, the performance of contention-based approach really depends on traffic load because collisions occur frequently when traffic load increases. As a result, a higher traffic load worsens the performance of contention-based MAC protocols.

4.5.2. Reservation-Based MAC Protocols. TRAMA [86] is a reservation-based MAC protocol which adopts the TDMA technique to minimize collisions and reduce the energy consumption. TRAMA separates the time into a *random access period* and a *scheduled access period*, both of which are composed of time slots. In the random access period, sensor nodes collect the information about the neighboring nodes using Neighbor Protocol (NP) and exchange their two-hop neighbor information and schedules through the Schedule Exchange Protocol (SEP). In addition, the Adaptive Election Algorithm (AEA) is used to determine the node that can transmit or receive data at a particular time slot in the scheduled period using the information obtained from NP and SEP. In the scheduled access period, sensor nodes send or receive data according to the schedules planned in the previous random access period. The other sensor nodes which have no activities in the scheduled period will enter the sleep mode to conserve energy.

Lu et al. [87] proposed an energy-efficient and low latency MAC protocol, called D-MAC, for tree-based data

gathering schemes in WSNs. D-MAC is an improved version of the Slotted Aloha protocol, where the time is divided into small slots and CSMA technique is adopted. In D-MAC, the awake/sleep schedule of sensor nodes are staggered based on their depth in the data gathering tree. In the best case, a packet can be delivered from source node to the sink node without any transmission latency. However, DMAC is not quite flexible. When the tree topology changes, the awake/sleep schedule of sensors needs to change also, resulting in a poor performance in terms of energy consumption.

PEDAMACS [88] is a TDMA-based MAC protocol which can be used for a multi-hop network environment. In PEDAMACS, the destination of all data packets generated by sensors is the same access point (or sink node). The sink is assumed to be powerful enough such that its transmission range can fully cover all sensor nodes in the monitoring region. In the topology collection phase, the CSMA technology is adopted so that sensor nodes can send their information to the sink. Upon receiving information from all sensors, the sink plans a global data report schedule and sends this schedule to all sensors. Furthermore, in order to cope with the collision problem, the collision-free scheduling of PEDAMACS is based on *the coloring of the original conflict graph*. In this graph coloring method, the *color* denotes the transmission time slot of nodes. Hence, no two adjacent nodes share the same color, avoiding the collision problem.

Several popular reservation-based MAC protocols [86–88] were presented for WSNs in this subsection. In contrast to contention-based MAC protocols, reservation-based approaches require knowledge of the network topology so that an active/sleep schedule can be planned. Moreover, in most reservation-based schemes, the time is usually divided into a number of small slots. Any sensor intending to transmit (receive) data packet is allocated a particular time slot and then transmits (receives) the data packet to (from) the receiver (sender) in that time slot. Based on this behavior, the collision problem can be improved significantly. Nevertheless, the reservation-based methods require strict time synchronization. If the time is not synchronized, the collisions will occur more frequently, further causing degradation in the performance of the reservation-based MAC protocols.

5. CPS Deployment Application Areas

This section surveys a few well-known CPS applications from different domains and highlights their key technologies. These applications can be categorized into the following categories: smart spaces, healthcare, emergency real-time systems, environmental monitoring and control as well as smart transportation.

5.1. Smart Spaces. We present a survey of several smart space applications [4, 92–94]. Based on such applications, many daily activities can be performed more intelligently and conveniently.

Chun et al. [92] proposed an agent-based self-adaptation architecture to create intelligent devices for smart space applications to ensure the reliability and predictability requirements in CPS designs. Their proposed architecture includes a self-adaptive robot which is equipped with sensors such as electronic compass, motor, web camera, and so forth. When the robot detects events via sensors, the self-adaptive system (which can be treated as the decision making system) executes the self-adaptation process to control the robot's behavior.

References [93, 94] considered the energy conservation issue in smart space applications. Han and Lim [93] designed and implemented a *smart home energy management system* (decision making system) using WSN technology. The designed system mainly consists of three components: *sensing infra*, *context-aware*, and *service management*. The sensing infra component is used to receive sensing data (such as temperature, noise level, and light intensity) from sensor nodes deployed in the smart space. The context-aware component is responsible for information analysis and provides the decision component with relevant information. The service management is a decision component which makes decisions to control appliances in the physical world. Depending on the design of the system, the energy usage in the smart space could be managed in an efficient way.

Byun and Park [94] developed a self-adapting intelligent system to make daily appliances more energy efficient and more intelligent. The developed system is composed of a *Self-adapting Intelligent Gateway* (SIG) and a *Self-adapting Intelligent Sensor* (SIS). The SIG, which can be regarded as decision making system, is responsible for several tasks, including appliance and sensor node management, service decisions, power/environmental information collection and analysis, provision of energy management services, and so forth. The SIS is used to collect situational information and provides the energy and environmental information for the SIG. As a result, the SIG could offer the adequate services based on the information from SIS when certain specific events occur.

5.2. Healthcare. Healthcare applications [95, 96] could acquire vital signs from medical sensors worn by patients or elders. The acquired data can later be used by some real-time decision making system.

Huang et al. [95] presented a healthcare monitoring architecture using WSN technology. The designed architecture is composed of three tiers: *sensor network*, *mobile computing network*, and *back-end network tiers*. In the sensor network tier, the *Wearable Sensor System* (WSS) and *Wireless Sensor Mote* (WSM) system are used to capture the vital signs of people and collect the environmental information inside the buildings, respectively. In the mobile computing network tier, the vital sign and environmental information are sent to the back-end network tier via mobile computing devices (such as PDA, smart phone and laptop). Finally, in the back-end network tier, the decision making system stores and analyzes the information received from mobile computing devices and offers application services.

López et al. [96] developed another healthcare platform, called LOBIN, which is also based on WSN technology. The LOBIN platform consists of four subsystems: *healthcare-monitoring*, *location*, *WSN*, and *management subsystems*. The healthcare-monitoring subsystem captures the vital signs of patients by the wearable sensors while the location subsystem aims to help patients acquire their location information. In addition, both vital signs and location information are sent to a management subsystem through the WSN subsystem. The management subsystem (decision making system) analyzes and stores the received information and later uses the information in the decision process.

5.3. Emergency Real-Time Systems. Emergency real-time systems [97–99] could not only help people avoid natural disasters (such as tsunami, volcanic eruption or mudslide) but also provide potential escape solutions for people. As a result, life will be safer and more secure.

Research efforts described in [97, 98] employed WSN technology to develop an emergency real-time navigation system, which could guide people to the safe areas when certain disasters occur. The basic idea behind the proposed solution in [97] is to select a subset of sensor nodes to construct a *skeleton graph* which contains fewer nodes and then offer people escape solutions based on the skeleton graph. Li et al. [98] used a road map system to help people discover an escape route. When a specific event occurs, the road map is periodically updated based on the current locations of the unsafe areas. As a result, for these disaster scenarios applying the solutions proposed in [97] or [98], people could send local queries to the nearby sensors via their mobile communication devices (such as PDA and smart phone) and obtain an escape route once some dangerous event occurs.

Casey et al. [99] developed an emergency system for tsunami detection and mitigation using WSN technology. Many sensor nodes are deployed over the coastal area and each of them falls into one of the three states, including *sensor*, *commander*, and *barrier*. Each sensor node is responsible for pressure information collection and sends the collected data to the commander node (which can be viewed as the decision making system). The commander node then selects a set of barrier nodes to reduce the impact of the wave in accordance with the information from the sensor nodes.

5.4. Environmental Monitoring and Control. Environmental monitoring helps to extend the human ability to understand the real world, and the *combination of virtual and reality to Internet of Things* [100]. WSNs have been effectively applied in military and civil applications covering areas such as target field imaging, intrusion detection, weather monitoring, security and tactical surveillance, distributed computing and control, and so on. In such a scenario of monitoring environment, users expect to obtain the information immediately when normal or unexpected events occurred and they can inquire about the data of interest. WSNs [101] also contribute toward making environmental monitoring more convenient and automated.

WSNs used in environmental monitoring involve a large number of low cost, low-power, small size, multi-node consisting of sensors, the use of IEEE802.15.4/ZigBee protocol, with each sensor having equipped with processor, memory, power supply, radio transceiver, and carry different sensing elements [5] to collect sensor data including temperature, humidity, pressure, air quality, wind speed, wind direction, rainfall, chemicals, and light intensity. The sensor node sensing environmental information transmits its sensor data via wireless communications using multi-hop transmission technology to the Sink node [102] which sends the sensor data to the external network. To achieve data management and remote access capabilities, users can access this sensor data over the Internet.

Environmental monitoring applications can be broadly classified into two categories namely, indoor and outdoor monitoring [103]. Indoor monitoring includes health monitoring, power monitoring, product address monitoring, factory logistics automation, civil structures deformations monitoring. Outdoor monitoring [104] includes chemical hazardous detection, habitat monitoring, traffic monitoring, earthquake detection, volcano eruption, flooding detection and weather forecasting. Tracing includes object, animal, human, and vehicle.

Environmental monitoring depends on wireless sensor nodes for its data, but the storage of a sensor node is small. Resources are very Limited, using WSN operating system such as TinyOS [105], Contiki [106]. These operating systems can make the sensor node run efficiently, but given its limited resource, there are still a lot of challenges [102] that need to be addressed such as power control, energy-efficient protocol, cost, reliable data transmission, and remote management.

For power control, communications between one sensor node and another need a lot of energy [107]: about 60% in listening idle even though 90% of the total energy is wasted in waking up the sensor node. To minimize energy, either a Mesh or a Route method is used to increase the network lifetime. In terms of convenience, because the installation of a sensor node is too complex, we need to develop easier modules that can be easily installed and maintained. IPv6 combined with WSNs can make the latter energy efficient. In terms of cost, because WSNs need many nodes, the cost of hardware becomes an important factor. In terms of reliability, when a sensor node transmits data, the data need to pass through a sequence nodes when real-time events happen or a middle node fault occurs. With remote management, if a node exists independently in a site, then that node will not be managed.

In [108], monitoring the volcano in Ecuador is achieved using acoustic sensors to collect volcano data. To extract high-fidelity data from such a WSN is challenging for two primary reasons. First, the radio links are lossy and frequently asymmetrical. Second, the low-cost crystal oscillators on the sensor nodes have low tolerances causing the clock rate to vary across the network. Much prior research efforts have focused on addressing these challenges [109, 110]. In [108], they developed a reliable data-collection protocol called Fetch using a tag and a routing tree to solve the

first problem, and chose the Flooding Time Synchronization Protocol (FTSP) to solve the clock rate problem.

5.5. Smart Transportation. Vehicular Sensor Networks (VSNs) [111–113] have been receiving a lot of attention recently. In VSNs, the vehicular sensors are attached to vehicles such as buses and cars. Intelligent Transportation Systems (ITS) improve road safety and convenience, manage vehicle traffic, and provide passengers with information. VSNs are considerably different from traditional WSNs environments. In fact, vehicular sensors are not affected by strict energy constrains and storage capabilities because sensors are embedded in vehicles. VSNs are distributed and self-organizing communication networks built up from moving vehicles, and are thus characterized by very high speeds and large-scale vehicular networks. Due to these special characteristics, VSNs have many challenging research issues resulting from the high mobility of vehicles, the wide range of relative speeds between vehicular nodes, and the real-time nature of applications. In this paper, we focus on challenging issues related to VSNs.

(1) Routing Protocols. In [114], the sensors equipped with GPS are placed on the roadside, and data is collected from vehicular sensors. The routing step can be divided into three phases. In the data requesting phase, the vehicle which needs data will send a request packet to a certain sensor node that is currently closest to the position of a vehicle. The sensor node that receives the request packet will retrieve the information of interest. In the data replication phase, the sensor node will flood the data packets to its neighbors, creating replicas of the data for its neighbors. In the data sharing phase, when data traffic increases dramatically due to many vehicles requesting for data at same time, the proposed method can send the information of interest to every vehicle. In [115], when a mobile sensor moves into the communication range of a road side sensor, the road side sensor detects this mobile sensor and sends a connection request to the mobile sensor. When a vehicular sensor is moving on a road segment where it is out of range of road side sensors, it will communicate with other cars.

(2) Data Dissemination. In [116], the authors proposed an algorithm that uses a grid-based hierarchical structure. When data is aggregated, data is then forwarded to layers of the hierarchical structure. The data aggregated will be disseminated to nodes in the lower layers. If a user wants the data of a small region, the system will provide the user with accurate data.

(3) Surveillance. In [117], the mobility of vehicles in highly dynamic and unpredictable network topologies lead to packet losses and distorted surveillance results. The authors proposed a method that is based on cooperative data sensing and used a compressing approach with zero inter-sensor collaboration and compression overhead based on sparse random projections. In [118], a theoretical model is introduced to analyze the communication costs of data transmissions in WSNs. A graph-based algorithm is proposed with a communication-cost graph used to depict the cost of data transmission and a modified Dijkstra's algorithm is used to find optimal solutions with reduced time complexity.

(4) Navigation. In [119], the authors proposed a method to address traffic congestion in large cities. They presented a dynamic navigation protocol for individual vehicles to find the shortest-time paths toward their destinations. There are proposed methods to reduce communication costs and support an error handling mechanism to deal with abnormal circumstances.

(5) Communication. In [120], a new opportunistic network approach where vehicles act as the communication infrastructure, furnishing low-cost asynchronous communications, variable delays, and limited bandwidth is presented. In [121], the method is a vehicle-density-based emergency broadcast scheme to solve the problem of receiver-oriented schemes. Two types of multi-hop broadcasting forwarder selection schemes for emergency broadcasting are proposed.

6. Integration of WSNs with CPSs

In CPS designs, the reliability and predictability are two important factors as we mentioned earlier in this paper. To ensure that those two important factors are fully supported, this work reviews the five fundamental WSN characteristics and surveys various CPS applications from different domains. Nonetheless, there are several challenges that need to be overcome so that CPSs can efficiently be implemented and deployed. These challenges include the *integration of appliances with different communication protocols*, *mobility of sensor node*, *remote access*, and *unachievable and unrealistic theoretical assumptions*. We describe a few of these challenges below.

(1) Integration of Appliances with Different Communication Protocols. Today, electronic devices execute different communication protocols, such as WiFi, Bluetooth, Zigbee, RF, infrared, and so forth. Indeed, some traditional devices have no wireless communication functions. As a result, the real-time information and status of devices cannot effectively be integrated and communicated. Similarly, different types of sensor nodes (such as iMote and MicaZ) might be also used in the same CPS application. The integration and interoperability of heterogeneous sensor nodes in CPSs remains a significant challenge.

(2) Mobility of Sensor Node. In the mobile WSN, many existing coverage schemes use the characteristic of sensor mobility to ensure both monitoring quality and network connectivity. However, in CPS applications, the sensor nodes might be embedded in the daily appliances such as PDA, smart phone and vehicles. Undoubtedly, the movement of appliances is determined by human rather than sensor nodes. In fact, both the monitoring quality and network

connectivity might not be guaranteed by existing coverage schemes. This lack of guarantee remains a challenge for future CPS designs.

(3) Remote Access. In most CPS applications such as those deployed in healthcare and emergency real-time systems, sensor nodes might require sending their readings to the decision making system via the Internet. Internet access availability may be an issue. This is because the WSN has so far been considered only as a standalone system and thus sensors did not require access to the Internet. Furthermore, the traditional Internet uses IPv4 technique which is unsuitable for WSNs due to the limited address space of IPv4. Today, there are a few research efforts [107, 122, 123] which have been using IPv6 technology on WSNs to address the limited address problem. However, to the best of our knowledge, this approach is still not mature for WSNs and future CPSs designs relying on WSNs will need to address this issue.

(4) Unachievable and Unrealistic Theoretical Assumptions. There are many well-known WSN schemes for various kinds of applications that have been proposed in the last decade. However, most of them make unachievable and unrealistic theoretical assumptions in the physical world, making it hard to design and build CPSs building practice. For example, the sensing range and communication range are usually assumed as a perfect disk. However, this assumption is obviously not adequate for the physical world since both of them are irregular in practice.

To successfully enable the design and deployment of CPSs that can leverage WSN technologies, the abovementioned challenges in this section have to be overcome. Otherwise, the real-time decision making system might not have all the available CPS inputs for timely decisions to be made, leading to possible performance degradation of CPS applications.

7. Conclusions

CPS designs show great promise in enabling human-to-human, human-to-object, and object-to-object interactions between the physical world and the virtual world. CPS applications have tremendous potential to improve safety, convenience, and comfort in our daily life. To ensure the reliability and predictability of CPSs, we need to be able to make real-time decisions using all available CPS inputs. We argue, in this work, that by leveraging WSN characteristics and its integration into CPS designs it is possible to provide timely CPS inputs. In addition, this paper presents a survey of several well-known CPS applications from different domains, including smart space, healthcare, emergency real-time system, environmental monitoring and control as well as smart transportation, and highlights their key technological features. Finally, we discuss some of the challenges that we still need to overcome by investigating innovative solutions that can enable seamless integration of heterogeneous devices, protocols, and design architectures with emerging CPS designs. Such solutions will help designers build more reliable and predictable CPSs in the future.

Acknowledgments

We thank the anonymous reviewers for their valuable comments which helped to improve the quality and presentation of this paper.

References

[1] E. A. Lee, "Cyber physical systems: design challenges," in *Proceedings of the 11th IEEE Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC '08)*, pp. 363–369, May 2008.

[2] E. A. Lee, "Cyber-physical systems—are computing foundations adequate?" in *Position Paper for NSF Workshop on Cyber-Physical Systems: Research Motivation, Techniques and Roadmap*, Austin, Tex, USA, October 2006.

[3] J. Wan, H. Yan, H. Suo, and F. Li, "Advances in cyber-physical systems research," *KSII Transactions on Internet and Information Systems*, vol. 5, no. 11, pp. 1891–1908, 2011.

[4] N. Correll, N. Arechiga, A. Bolger et al., "Building a distributed robot garden," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '09)*, pp. 1509–1516, October 2009.

[5] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[6] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next century challenges: mobile networking for 'Smart Dust'," in *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (ACM/IEEE MobiCom '99)*, August 1999.

[7] S. Hollar, *COTS dust*, M.S. thesis, University of California, Berkeley, Calif, USA, 2002.

[8] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, pp. 364–369, April 2005.

[9] J. Beutel, O. Kasten, and M. Ringwald, "Poster abstract: BTnodes—a distributed platform for sensor nodes," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*, pp. 292–293, November 2003.

[10] L. Nachman, R. Kling, R. Adler, J. Huang, and V. Hummel, "The Intel® mote platform: a Bluetooth∗-based sensor network for industrial monitoring," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, pp. 437–442, April 2005.

[11] R. Adler, M. Flanigan, J. Huang et al., "Demo abstract: intel mote 2: an advanced platform for damanding sensor network applications," in *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys '05)*, November 2005.

[12] P. Levis and D. Gay, *TinyOS Programming*, Cambridge University Press, New York, NY, USA, 2009.

[13] E. Biagioni and G. Sasaki, "Wireless sensor placement for reliable and efficient data collection," in *Proceedings of the IEEE Hawaii International Conference on System Sciences (IEEE HICSS '03)*, January 2003.

[14] J. O'Rourke, *Art Gallery Theorem and Algorithms*, Oxford University Press, New York, NY, USA, 1987.

[15] L. Schwiebert, S. K. S. Gupta, and J. Weinmann, "Research challenges in wireless networks of biomedical sensors," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pp. 151–165, July 2001.

[16] W. Li and C. G. Cassandras, "A minimum-power wireless sensor network self-deployment scheme," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '05)*, pp. 1897–1902, March 2005.

[17] B. Cărbunar, A. Grama, J. Vitek, and O. Cărbunar, "Redundancy and coverage detection in sensor networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 1, pp. 94–128, 2006.

[18] H. Gupta, Z. Zhou, S. R. Das, and Q. Gu, "Connected sensor cover: self-organization of sensor networks for efficient query execution," *IEEE/ACM Transactions on Networking*, vol. 14, no. 1, pp. 55–67, 2006.

[19] C. Y. Chang and H. R. Chang, "Energy-aware node placement, topology control and MAC scheduling for wireless sensor networks," *Computer Networks*, vol. 52, no. 11, pp. 2189–2204, 2008.

[20] S. Chellappan, X. Bai, B. Ma, D. Xuan, and C. Xu, "Mobility limited flip-based sensor networks deployment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 2, pp. 199–211, 2007.

[21] N. Heo and P. K. Varshney, "Energy-efficient deployment of intelligent mobile sensor networks," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.*, vol. 35, no. 1, pp. 78–92, 2005.

[22] A. Sekhar, B. S. Manoj, and C. Siva Ram Murthy, "Dynamic coverage maintenance algorithms for sensor networks with limited mobility," in *Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications (PerCom '05)*, pp. 51–60, March 2005.

[23] G. Wang, G. Cao, and T. F. La Porta, "Movement-assisted sensor deployment," *IEEE Transactions on Mobile Computing*, vol. 5, no. 6, pp. 640–652, 2006.

[24] M. A. Batalin and G. S. Sukhatme, "Efficient exploration without localization," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2714–2719, September 2003.

[25] M. A. Batalin and G. S. Sukhatme, "Coverage, exploration and deployment by a mobile robot and communication network," *Telecommunication Systems, Special Issue on Wireless Sensor Networks*, vol. 26, no. 2–4, pp. 181–196, 2004.

[26] M. A. Batalin and G. S. Sukhatme, "The design and analysis of an efficient local algorithm for coverage and exploration based on sensor network deployment," *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 661–675, 2007.

[27] Y. C. Wang, C. C. Hu, and Y. C. Tseng, "Efficient deployment algorithms for ensuring coverage and connectivity of wireless sensor networks," in *Proceedings of the 1st International Conference on Wireless Internet (WICON '05)*, pp. 114–121, July 2005.

[28] C. Y. Chang, J. P. Sheu, Y. C. Chen, and S. W. Chang, "An obstacle-free and power-efficient deployment algorithm for wireless sensor networks," *IEEE Transactions on Systems, Man, and Cybernetics Part A*, vol. 39, no. 4, pp. 795–806, 2009.

[29] C. Y. Chang, C. T. Chang, Y. C. Chen, and H. R. Chang, "Obstacle-resistant deployment algorithms for wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 6, pp. 2925–2941, 2009.

[30] G. Dommety and R. Jain, "Potential networking applications of global positioning systems (GPS)," Tech. Rep. TR-24, The Ohio State University, 1996.

[31] B. W. Parkinson and S. W. Gilbert, "NAVSTAR: global positioning system-ten years later," *Proceedings of the IEEE*, vol. 71, no. 10, pp. 1177–1186, 1983.

[32] Y. Weng, W. Xiao, and L. Xie, "Total least squares method for robust source localization in sensor networks using TDOA measurements," *International Journal of Distributed Sensor Networks*, vol. 2011, Article ID 172902, 8 pages, 2011.

[33] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "Cricket location-support system," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM '00)*, pp. 32–43, August 2000.

[34] D. Niculescu and B. Nath, "Ad Hoc positioning system (APS) using AoA," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM '03)*, March 2003.

[35] J. Hightower, G. Boriello, and R. Want, "SpotON: an indoor 3D location sensing technology based on RF signal strength," UW CSE Technical Report, University of Washington, 2002.

[36] P. Bahl and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM '00)*, Tel Aviv, Israel, March 2000.

[37] N. Patwari and A. O. Hero, "Using proximity and quantized RSS for sensor localization in wireless networks," in *Proceedings of the 2nd ACM International Workshop on Wireless Sensor Networks and Applications (WSNA '03)*, pp. 20–29, September 2003.

[38] A. Savvides, C. C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in Ad-Hoc networks of sensors," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pp. 166–179, July 2001.

[39] D. Niculescu and B. Nath, "DV based positioning in Ad Hoc networks," *Telecommunication Systems*, vol. 22, no. 1–4, pp. 267–280, 2003.

[40] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, "Range-free localization schemes for large scale sensor networks," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom '03)*, pp. 81–95, September 2003.

[41] K. F. Ssu, C. H. Ou, and H. C. Jiau, "Localization with mobile anchor points in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 54, no. 3, pp. 1187–1197, 2005.

[42] C. H. Ou, K. F. Ssu, and H. C. Jiau, "Range-free localization with aerial anchors in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2, no. 1, pp. 1–21, 2006.

[43] Y. Kwon and G. Agha, "Passive localization: large size sensor network localization based on environmental events," in *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN '08)*, pp. 3–14, April 2008.

[44] M. Rudafshani and S. Datta, "Localization in wireless sensor networks," in *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks (IPSN '07)*, pp. 51–60, April 2007.

[45] V. Vivekanandan and V. W. S. Wong, "Concentric anchor beacon localization algorithm for wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 5, pp. 2733–2744, 2007.

[46] K. Kim and W. Lee, "MBAL: a mobile beacon-assisted localization scheme for wireless sensor networks," in *Proceedings of the 16th International Conference on Computer Communications and Networks (ICCCN '07)*, pp. 57–62, August 2007.

[47] G. Teng, K. Zheng, and W. Dong, "Adapting mobile beacon-assisted localization in wireless sensor networks," *Sensors*, vol. 9, no. 4, pp. 2760–2779, 2009.

[48] B. Xiao, H. Chen, and S. Zhou, "Distributed localization using a moving beacon in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 5, pp. 587–600, 2008.

[49] T. V. Srinath, "Localization in resource constrained sensor networks using a mobile beacon with in-ranging," in *Proceedings of the IFIP International Conference on Wireless and Optical Communications Networks*, April 2006.

[50] J. P. Sheu, W. K. Hu, and J. C. Lin, "Distributed localization scheme for mobile sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 4, pp. 516–526, 2010.

[51] A. Galstyan, B. Krishnamachari, K. Lerman, and S. Pattem, "Distributed online localization in sensor networks using a moving target," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN '04)*, pp. 61–70, April 2004.

[52] C. T. Chang, C. Y. Chang, and C. Y. Lin, "Anchor-guiding mechanism for beacon-assisted localization in wireless sensor networks," *IEEE Sensors Journal*, no. 99, p. 1, 2011.

[53] C. Y. Chang, C. Y. Lin, and C. T. Chang, "Tone-based localization for distinguishing relative locations in wireless sensor networks," *IEEE Sensors Journal*, no. 99, p. 1, 2011.

[54] P. Bergamo and G. Mazzini, "Localization in sensor networks with fading and mobility," in *Proceedings of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '02)*, pp. 750–754, September 2002.

[55] X. Li, H. Frey, N. Santoro, and I. Stojmenovic, "Strictly localized sensor self-deployment for optimal focused coverage," *IEEE Transactions on Mobile Computing*, vol. 10, no. 11, pp. 1520–1533, 2011.

[56] G. J. Fan and S. Y. Jin, "Coverage problem in wireless sensor network: a survey," *Journal of Networks*, vol. 5, no. 9, pp. 1033–1040, 2010.

[57] C. F. Huang and Y. C. Tseng, "The coverage problem in a wireless sensor network," *Mobile Networks and Applications*, vol. 10, no. 4, pp. 519–528, 2005.

[58] S. Kumar, T. H. Lai, and J. Balogh, "On k-coverage in a mostly sleeping sensor network," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking (MobiCom '04)*, pp. 144–158, October 2004.

[59] L. Lin and H. Lee, "Brief announcement: distributed algorithms for dynamic coverage in sensor networks," in *Proceedings of the 26th Annual ACM Symposium on Principles of Distributed Computing (PODC '07)*, pp. 392–393, August 2007.

[60] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava, "Coverage problems in wireless Ad-Hoc sensor networks," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM '01)*, April 2001.

[61] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar, "Reliable density estimates for coverage and connectivity in thin strips of finite length," in *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking (MobiCom '07)*, pp. 75–86, September 2007.

[62] A. Chen, S. Kumar, and T. H. Lai, "Designing localized algorithms for barrier coverage," in *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking (MobiCom '07)*, pp. 63–74, September 2007.

[63] S. Kumar, T. H. Lai, and A. Arora, "Barrier coverage with wireless sensors," in *Proceedings of the 11th Annual International Conference on Mobile Computing and Networking (MobiCom '05)*, pp. 284–298, September 2005.

[64] C. Shen, W. Cheng, X. Liao, and S. Peng, "Barrier coverage with mobile sensors," in *Proceedings of the 9th International Symposium on Parallel Architectures, Algorithms and Networks (I-SPAN '08)*, pp. 99–104, May 2008.

[65] M. Li, W. Cheng, K. Liu, Y. He, X. Li, and X. Liao, "Sweep coverage with mobile sensors," *IEEE Transactions on Mobile Computing*, vol. 10, no. 11, pp. 1534–1545, 2011.

[66] A. Bereketli and O. B. Akan, "Event-to-sink directed clustering in wireless sensor networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '09)*, April 2009.

[67] W. Alsalih, H. Hassanein, and S. Akl, "Routing to a mobile data collector on a predefined trajectory," in *Proceedings of the IEEE International Conference on Communications (ICC '09)*, June 2009.

[68] Z. Zhang, M. Ma, and Y. Yang, "Energy-efficient multihop polling in clusters of two-layered heterogeneous sensor networks," *IEEE Transactions on Computers*, vol. 57, no. 2, pp. 231–245, 2008.

[69] F. Wang and J. Liu, "Networked wireless sensor data collection: issues, challenges, and approaches," *IEEE Commmunications Surveys and Tutorials*, vol. 13, no. 4, pp. 673–687, 2011.

[70] W. Liang and Y. Liu, "Online data gathering for maximizing network lifetime in sensor networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 1, pp. 2–11, 2007.

[71] C. Wang, C. Jiang, S. Tang, and X. Li, "SelectCast: scalable data aggregation scheme in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 99, p. 1, 2012.

[72] O. D. Incel, A. Ghosh, B. Krishnamachari, and K. Chintalapudi, "Fast data collection in tree-based wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 1, pp. 86–99, 2012.

[73] H. Arjmandi and F. Lahouti, "Resource optimized distributed source coding for complexity constrained data gathering wireless sensor networks," *IEEE Sensors Journal*, vol. 11, no. 9, pp. 2094–2101, 2011.

[74] C. Tan, J. Zou, M. Wang, and H. Xiong, "Correlated data gathering on dynamic network coding policy and opportunistic routing in wireless sensor network," in *Proceedings of the IEEE International Conference on Communications (IEEE ICC '11)*, June 2011.

[75] S. Jain, R. C. Shah, W. Brunette, G. Borriello, and S. Roy, "Exploiting mobility for energy efficient data collection in wireless sensor networks," *Mobile Networks and Applications*, vol. 11, no. 3, pp. 327–339, 2006.

[76] M. Zhao, M. Ma, and Y. Yang, "Efficient data gathering with mobile collectors and space-division multiple access technique in wireless sensor networks," *IEEE Transactions on Computers*, vol. 60, no. 3, pp. 400–417, 2011.

[77] G. Xing, M. Li, T. Wang, W. Jia, and J. Huang, "Efficient rendezvous algorithms for mobility-enabled wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 1, pp. 47–60, 2012.

[78] X. Fei, A. Boukerche, and R. Yu, "An efficient Markov decision process based mobile data gathering protocol for wireless sensor networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '11)*, pp. 1032–1037, March 2011.

[79] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM '02)*, June 2002.

[80] LAN MAN Standards Committee of the IEEE Computer Society, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-1999*, IEEE, 1999.

[81] S. Singh and C. S. Raghavendra, "PAMAS: power aware multi-access protocol with signaling for Ad Hoc networks," *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 3, pp. 5–26, 1998.

[82] T. Van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*, pp. 171–180, November 2003.

[83] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, pp. 95–107, November 2004.

[84] C. C. Enz, A. El-Hoiydi, J. D. Decotignie, and V. Peiris, "WiseNET: an ultralow-power wireless sensor network solution," *Computer*, vol. 37, no. 8, pp. 62–70, 2004.

[85] A. El-Hoiydi, "Spatial TDMA and CSMA with preamble sampling for low power Ad Hoc wireless sensor networks," in *Proceedings of the IEEE International Symposium on Computers and Communications (IEEE ISCC '02)*, July 2002.

[86] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," *Wireless Networks*, vol. 12, no. 1, pp. 63–78, 2006.

[87] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An adaptive energy-efficient and low-latency MAC for data gathering in wireless sensor networks," in *Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS '04)*, pp. 3091–3098, April 2004.

[88] S. C. Ergen and P. Varaiya, "PEDAMACS: power efficient and delay aware medium access protocol for sensor networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 7, pp. 920–930, 2006.

[89] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: a media access protocol for wireless LAN's," in *Proceedings of the ACM Conference on Communications Architectures, Protocols and Applications (ACM SIGCOMM '94)*, October 1994.

[90] M. Stemm and R. H. Katz, "Measuring and reducing energy consumption of network interfaces in hand-held devices," *IEICE Transactions on Communications*, vol. 80, no. 8, pp. 1125–1131, 1997.

[91] P. Karn, "MACA: a new channel access method for packet radio," in *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, Ontario, Canada, Sepetember 1990.

[92] I. Chun, J. Park, H. Lee, W. Kim, S. Park, and E. Lee, "An agent-based self-adaptation architecture for implementing smart devices in smart space," *Telecommunication Systems*, 2011.

[93] D. M. Han and J. H. Lim, "Design and implementation of smart home energy management systems based on ZigBee," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1417–1425, 2010.

[94] J. Byun and S. Park, "Development of a self-adapting intelligent system for building energy saving and context-aware smart services," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 1, pp. 90–98, 2011.

[95] Y. M. Huang, M. Y. Hsieh, H. C. Chao, S. H. Hung, and J. H. Park, "Pervasive, secure access to a hierarchical sensor-based healthcare monitoring architecture in wireless heterogeneous networks," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 4, pp. 400–411, 2009.

[96] G. López, V. Custodio, and J. I. Moreno, "LOBIN: e-textile and wireless-sensor-network-based platform for healthcare monitoring in future hospital environments," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 6, pp. 1446–1458, 2010.

[97] C. Buragohain, D. Agrawal, and S. Suri, "Distributed navigation algorithm for sensor networks," in *Proceedings of the IEEE International Conference on Computer Communications (IEEE INFOCOM '06)*, April 2006.

[98] M. Li, Y. Liu, J. Wang, and Z. Yang, "Sensor network navigation without locations," in *Proceedings of the IEEE International Conference on Computer Communications (IEEE INFOCOM '9)*, April 2009.

[99] K. Casey, A. Lim, and G. Dozier, "A sensor network architecture for Tsunami detection and response," *International Journal of Distributed Sensor Networks*, vol. 4, no. 1, pp. 28–43, 2008.

[100] N. Gershenfeld, R. Krikorian, and D. Cohen, "The internet of things," *Scientific American*, vol. 291, no. 4, pp. 76–81, 2004.

[101] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[102] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer Networks*, vol. 51, no. 4, pp. 921–960, 2007.

[103] T. Arampatzis, J. Lygeros, and S. Manesis, "A survey of applications of wireless sensors and wireless sensor networks," in *Proceedings of the 13th Mediterranean Conference on Control and Automation (MED '05)*, pp. 719–724, June 2005.

[104] L. M. L. Oliveira and J. J. P. C. Rodrigues, "Wireless sensor networks: a survey on environmental monitoring," *Journal of Communications*, vol. 6, no. 2, pp. 143–151, 2011.

[105] TinyOS, July 2010, http://www.tinyos.net/.

[106] Contiki, July 2010, http://www.sics.se/contiki/.

[107] J. W. Hui and D. E. Culler, "IPv6 in low-power wireless networks," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1865–1878, 2010.

[108] G. Werner-Allen, K. Lorincz, M. Welsh et al., "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing*, vol. 10, no. 2, pp. 18–25, 2006.

[109] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*, pp. 14–27, November 2003.

[110] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *Proceedings of the International ACM Conference on Embedded Networked Sensor Systems (ACM SenSys '04)*, November 2004.

[111] G. Karagiannis, O. Altintas, and E. Ekici, "Vehicular networking: a survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE Communications Surveys and Tutorials*, vol. 13, no. 4, pp. 584–616, 2011.

[112] U. Lee, B. Zhou, M. Gerla, E. Magistretti, P. Bellavista, and A. Corradi, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *IEEE Wireless Communications*, vol. 13, no. 5, pp. 52–57, 2006.

[113] X. Su, "A comparative survey of routing protocol for vehicular sensor networks," in *Proceedings of the IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS '10)*, pp. 311–316, June 2010.

[114] K. W. Lim, W. S. Jung, and Y. B. Ko, "Multi-hop data dissemination with replicas in vehicular sensor networks," in *Proceedings of the 67th IEEE Vehicular Technology Conference-Spring (VTC '08)*, pp. 3062–3066, May 2008.

[115] F. Kong and J. Tan, "A collaboration-based hybrid vehicular sensor network architecture," in *Proceedings of the International IEEE Conference on Information and Automation (ICIA '08)*, pp. 584–589, June 2008.

[116] H. W. Tsai, T. S. Chen, and S. K. Lin, "Dissemination of data aggregation in vehicular Ad Hoc Networks," in *Proceedings of the 10th International Symposium on Pervasive Systems, Algorithms, and Networks (I-SPAN '09)*, pp. 625–630, December 2009.

[117] X. Yu, H. Zhao, L. Zhang, S. Wu, B. Krishnamachari, and V. O. K. Li, "Cooperative sensing and compression in vehicular sensor networks for urban monitoring," in *Proceedings of the IEEE International Conference on Communications (IEEE ICC '10)*, May 2010.

[118] W. Li, E. Chan, M. Hamdi, S. Lu, and D. Chen, "Communication cost minimization in wireless sensor and actor networks for road surveillance," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 2, pp. 618–631, 2011.

[119] W. Chen, S. Zhu, and D. Li, "VAN: vehicle-assisted shortest-time path navigation," in *Proceedings of the 7th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS '10)*, pp. 442–451, November 2010.

[120] J. N. Isento, J. A. Dias, J. J. P. C. Rodrigues, M. Chen, and K. Lin, "Performance assessment of aggregation and de-aggregation algorithms for vehicular delay-tolerant networks," in *Proceedings of the IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (IEEE MASS '11)*, October 2011.

[121] Y. T. Tseng, R. H. Jan, C. Chen, C. F. Wang, and H. H. Li, "A vehicle-density-based forwarding scheme for emergency message broadcasts in VANETs," in *Proceedings of the 7th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS '10)*, pp. 703–708, November 2010.

[122] M. Durvy, J. Abeille, P. Wetterwald et al., "Poster abstract: making sensor networks IPv6 ready," in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (ACM SenSys '08)*, November 2008.

[123] A. Leonardi, S. Palazzo, F. Scoto, and S. Signorello, "Enabling remote access to a wireless sensor network by exploiting IPv6 capabilities," in *Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC '11)*, July 2011.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 658724, 7 pages
doi:10.1155/2012/658724

Research Article

Noninvasive Wireless Sensor PFMT Device for Pelvic Floor Muscle Training

Jui-Fa Chen,[1] Huann-Cheng Horng,[2] Wei-Chuan Lin,[3] and Kun-Hsiao Tsai[1]

[1] *Department of Information Engineering and Computer Science, Tamkang University, Tamshui, New Taipei City 25137, Taiwan*
[2] *Department of Obstetrics and Gynecology, Taipei Veterans General Hospital, Taipei 11217, Taiwan*
[3] *Department of Information Technology, Takming College, Taipei 11451, Taiwan*

Correspondence should be addressed to Jui-Fa Chen, alpha@mail.tku.edu.tw

Received 14 December 2011; Accepted 16 March 2012

Academic Editor: Chih-Yung Chang

Urinary incontinence is a common problem among adults. Studies have shown up to 70% improvement in stress incontinence symptoms following appropriately performed pelvic floor exercise. This improvement is evident across all age groups. In this study, we cooperated with a doctor of the Department of Obstetrics and Gynecology, Taipei Veterans General Hospital, Taiwan. We developed a noninvasive device with the purpose of helping patients commence and perform pelvic floor muscle training (PFMT). This device consists of a PFMT device, an Arduino control board, a force sensor, a Bluetooth device, and an SD card. The objectives of this study are to train patients to inhibit detrusor contraction voluntarily and to contract periurethral muscles selectively. The system records and analyzes sensor data and provides voice prompts during PFMT exercise for patients at hospitals or their homes. Meanwhile, it tracks patients' PFMT exercise at home, and doctors can contact patients for additional visitation(s) if necessary.

1. Introduction

Urinary incontinence in women is a common, distressing, and costly health problem. Urinary incontinence often causes a loss of self-esteem, social isolation, and restriction of normal activity [1–4].

Pelvic floor muscle exercises, initiated by Kegel in 1948, strengthen the tone and contraction of periurethral and pelvic floor muscles. Pelvic floor muscle training (PFMT) or Kegel exercise has been found effective in reducing stress incontinence and also urge incontinence [5–8]. Studies have shown up to 70% improvement in stress incontinence symptoms following appropriately performed pelvic floor exercise. This improvement is evident across all age groups [9, 10].

However, several issues require addressing: (a) there is no doctor or continence nurse supervision and patients cannot determine whether PFMT is done correctly, at home, and (b) to obtain results from rehabilitation, doctors must wait at least one month for a patient to revisit. If problems exist in the patient's rehabilitation process, they will delay the diagnosis and treatment schedule [11, 12].

The aim of this study is to rehabilitate the pelvic musculature. We cooperated with Dr. Horng, a doctor of the Department of Obstetrics and Gynecology, Taipei Veterans General Hospital, Taiwan. We developed a non-invasive device with the purpose of helping patients perform PFMT. This device consists of a PFMT device, an Arduino control board, a force sensor, a Bluetooth device, and an SD card.

The objectives of this study are to (a) train patients to inhibit detrusor contraction voluntarily and contract periurethral muscles selectively, (b) record/analyze sensor data and also provide voice prompts during PFMT exercise for patients at hospitals and home, and (c) track patients at home during the PFMT process, so that doctors can monitor patients and initiate additional visitations, if necessary.

With this system, doctors can keep track of patients' courses of treatment, and patients can receive better medical care.

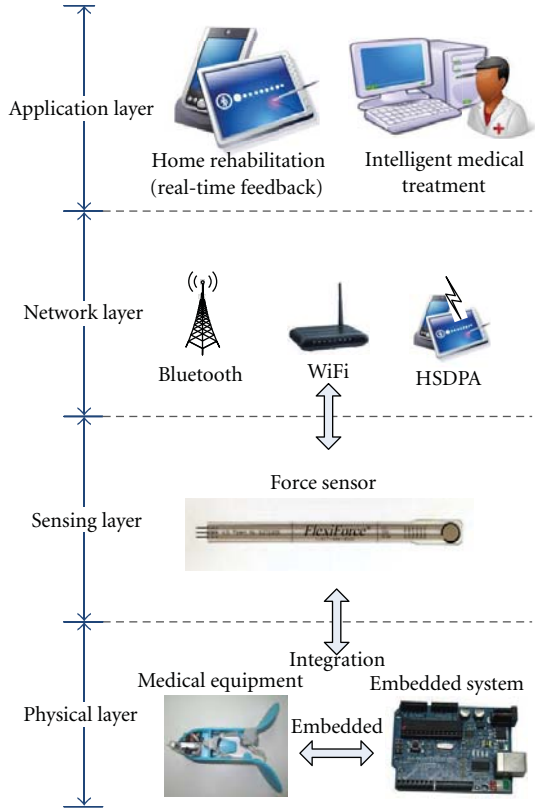


Figure 1: The architecture of the non-invasive wireless sensor PFMT device.

2. Materials and Methods

In this study, our system architecture is divided into four layers, the physical layer, sensing layer, network layer, and application ayer. The physical layer includes medical equipment and an embedded system. We integrated an embedded system into medical equipment to install sensors. The sensing layer consists of sensors used to sense the rehabilitation process of patients. The network layer uses wireless transmission technology to transfer data. Finally, the application layer, including the application programs that provide feedback to patients during the rehabilitation process, analyzes and tracks patients' rehabilitation for doctors. The architecture is shown in Figure 1.

Figure 2: The noninvasive wireless sensor PFMT device.

2.1. Physical Layer. We use a PFMT device, consisting of a U-shaped spring, which makes it easier for patients to use the non-invasive device for PFMT rehabilitation. The non-invasive wireless sensor PFMT device consists of a force sensor connected to an Arduino control board. The force sensor was placed inside the PFMT device and is used to obtain measurements of the contraction force of pelvic floor musculature, as illustrated in Figure 2.

2.2. Sensing Layer. We use force sensors to detect the patient's pelvic floor muscle contraction, which determine whether the PFMT rehabilitation of the patient is correct or not. Using the PFMT device shown in Figure 3, the process of pelvic floor muscle training by way of clamped PFMT device (used between the thighs) and variations in contraction force of pelvic floor musculature can be accurately measured.

2.3. Network Layer. The PFMT device includes a Bluetooth device that is used to transmit sensor data to a mobile device, such as a smartphone or a tablet, for analysis and feedback; an appropriate message will prompt patients on what steps should be taken after device usage. The rehabilitation data of a patient can be sent through mobile device directly to hospital servers. In this way, the doctor can keep track of patient rehabilitation and provide the appropriate

Figure 3: The pelvic floor muscle training with PFMT device.

treatment(s). If the Bluetooth device cannot successfully link to a smartphone, tablet, or desktop computer, an SD card is can be used in the PFMT device to store sensor data. When patients return to the hospital, doctors can obtain sensor data from the SD card and analyze the PFMT exercises done by patients.

2.4. Application Layer. The application layer is divided into two subsystems. One is the client system, and the other is the server system. The client system is a home rehabilitation system, whereas the server system, which is an intelligent medical treatment system, is built at the hospital. The detailed descriptions are as follows.

(1) Home rehabilitation system. The home rehabilitation system analyzes sensor data from the network layer and transmits responses to the patient via text message, audio message, video message, vibrate, and/or other multimedia. In this way, the patient can determine whether his/her actions are correct or not.

(2) Intelligent medical treatment system. Doctors can track and observe patients' PFMT exercises done at home by way of the intelligent medical treatment system. This system receives sensor data of rehabilitation exercise processes from the home rehabilitation system. Using the sensor data, this system analyzes patients' rehabilitation processes includes physical conditions and rehabilitation results. The features include (a) tracking whether patients perform daily PFMT exercise assigned by doctors, (b) analyzing whether or not the PFMT exercise process is correct, and (c) the adjusting of the rehabilitation plan depending on patient rehabilitation.

3. Data Processing and Analysis

First, patients performed PFMT rehabilitation exercise regimes supervised by doctors, specialist physiotherapists, and/or continence nurses at a hospital. We collected the sensor data and proposed a quantization method. The aforementioned method converts sensor signals into quantifiable data. Furthermore, we used statistical k-means clustering [13–16] and the k-nearest neighbor algorithm (k-NN) [17, 18] to analyze data. This provides important information for helping patients execute PFMT exercise and also assists doctors in medical care.

3.1. Data Collection. In accordance to doctor instructions, patients commence performing PFMT exercise. If necessary, doctors may use electrical stimulation to help patients understand how to correctly complete pelvic floor muscle contractions.

Two experimental examples of PFMT exercises are shown in Figures 4(a) and 4(b); the force of pelvic floor muscles is measured with respect to the force unit in pounds, as shown on y-axis. Figure 4(a) shows the signals of a patient's PFMT exercise, but the exercise results are incorrect. The force is strong. However, it is from thighs, not from pelvic floor muscle. Figure 4(b) shows the result after the patient correctly performs PFMT exercise.

3.2. Quantization. To analyze data of patient rehabilitation, we proposed quantization methods to convert sensor data into quantifiable data. To judge the correctness of a patient's exercise, we needed to know the force and duration of pelvic floor muscle contraction. The standard deviation is used to evaluate the stability of the patient's movements as shown in Figures 5 and 6.

We denoted t_i as the time of received i_{th} force sensor's data and f_i is the force value of i_{th} force sensor data. The duration T_k can be calculated by the subtraction of t_a and t_b, which is the k_{th} contraction time from time t_a to time t_b. The force value of k_{th} pelvic floor muscle contraction is F_k. The formulas are as follows.

Duration of a pelvic floor muscle contraction:

$$T_k = t_b - t_a. \tag{1}$$

Force of a pelvic floor muscle contraction:

$$F_k = \frac{\sum_{i=a}^{b-1}((f_i + f_{i+1})(t_{i+1} - t_i))/2}{T_k}. \tag{2}$$

We then convert data into a Cartesian coordinate system. Figure 7 shows the relationship of duration and force of pelvic floor muscle contraction motions in a PFMT exercise. A point with coordinate (X, Y) denotes the duration and force (T_k and F_k, resp.) of a contraction motion. A doctor can determine the overall status of a PFMT exercise from Figure 7 and also help patients improve upon their rehabilitation.

In order to allow doctors evaluate the quality of patient PFMT exercises, we provide a standard deviation force of each pelvic floor muscle contraction. In this way, doctors can review the detailed status of each contraction in a PFMT exercise and, if necessary, provide additional medical assistance to help patients improve their contraction motion. The formula is shown below.

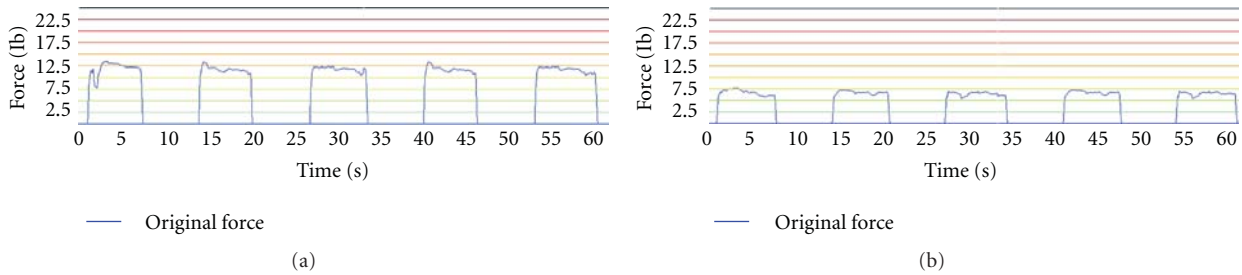


Figure 4: (a) Signals of an incorrect PFMT exercise, (b) Signals of a correct PFMT exercise.

Standard deviation force of a pelvic floor muscle contraction:

$$\mathrm{SDF}_k = \frac{\sum_{i=a}^{b} |f_i - F_k|}{T_k}. \tag{3}$$

Figure 8 illustrates the standard deviation of pelvic floor muscle contraction motions in a PFMT exercise. The duration and standard deviation (T_k, SDF_k) are represented by coordinate (X, Y).

3.3. Analysis. Because hospital patients can obtain assistance from doctors and nurses to do PFMT exercises correctly, much helpful pelvic floor muscle contraction sensor data can be collected by the PFMT device. Differences in patients' ages, genders, body types, and recovery situations are reflected in the quantized contraction sensor data, reinforcing the necessity to analyze data according to the individual.

After quantization, we can determine what constitutes correct contraction sensor data, such as the high-density cluster in the lower-right corner of Figure 7. In statistics and data mining, k-means clustering is a method used for cluster analysis. Thus, k-means clustering is used in this study to classify sensor data as correct or incorrect in performance of a contraction. The operations are formulated in (4) through (5) and are listed as followslinebreak [19, 20].

Given an initial set of k-means (m_1, m_2,..., m_k), the algorithm proceeds by alternating between formula (4) and formula (5):

$$C_i^{(t)} = \left\{ X_j : \left\| X_j - m_i^{(t)} \right\| \leqq \left\| X_j - m_{i^*}^{(t)} \right\| \quad \forall i^* = 1, 2, \ldots, k \right\}, \tag{4}$$

$$m_i^{(t+1)} = \frac{1}{\left| C_i^{(t)} \right|} \sum_{X_j \in C_i^{(t)}} X_j. \tag{5}$$

X_j is a point in the Cartesian coordinate system, C_i indicates the k_{th} cluster (C_1, C_2, ..., C_k), and t is the t_{th} calculation. The algorithm is considered to have converged when C_i no longer exhibits change and when, finally, we can obtain k-centroids of observations in the cluster. Doctors can provide guidelines for patient physical therapy based on the characteristics of each cluster.

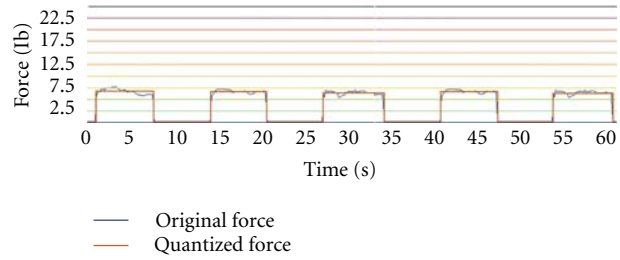


Figure 5: Original force and quantized force of five pelvic floor muscle contraction motions.

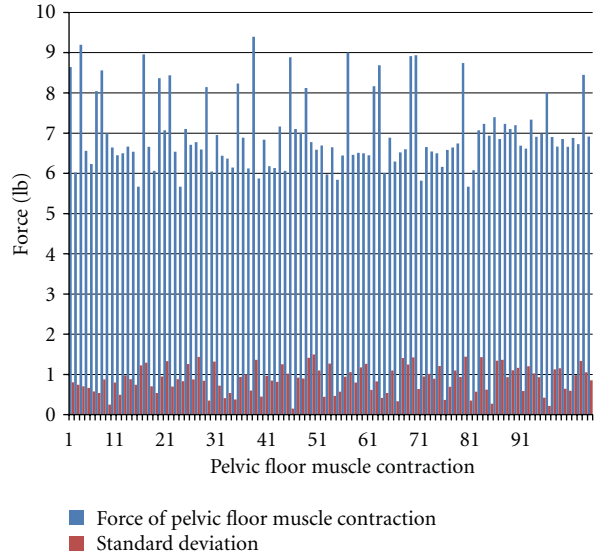


Figure 6: Quantized force and the standard deviations of force in a PFMT exercise.

3.4. Classification. According to the results of k-means classification, we use k-NN analysis for the sensor data of pelvic floor muscle contractions when patients perform PFMT exercises at home. K-NN can classify that a particular motion belongs to particular cluster, according to the characteristics of the cluster and doctor guidelines; we use text, audio, video, voice, and/or vibration feedback to assist patients in understanding the correctness of their PFMT exercise.

We can represent our data set containing P clusters C_1, C_2,..., C_P. A vector $W = \{W_1, W_2, \ldots, W_P\}$ with length P is used, listing the output value W_i for the weight of cluster C_i.

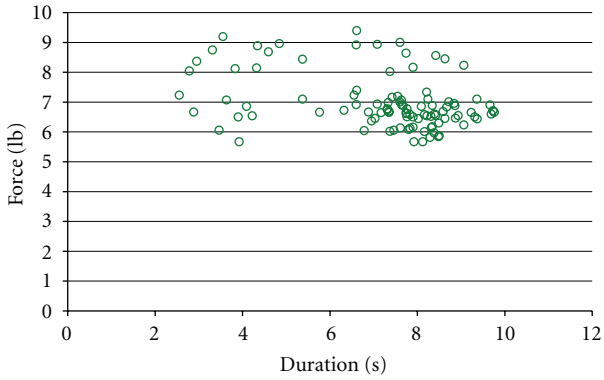


Figure 7: The relationship of duration and quantized force of pelvic floor muscle contraction motions in a PFMT exercise.

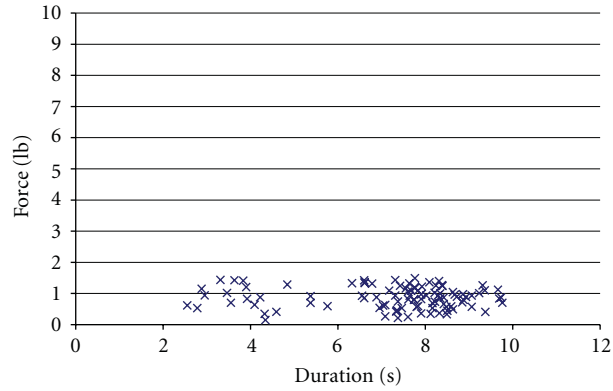


Figure 8: The standard deviation of pelvic floor muscle contraction motions in a PFMT exercise.

k-NN [21, 22] can be run in these steps.

(1) Store the output value of the k-nearest neighbors to a query contraction motion q in vector $R = \{R_1, R_2, \ldots, R_k\}$ by repeating the following loop k-times.

(a) $R \leftarrow X = \{X_i | \min(d(q, X_1), \ldots, d(q, X_P)), X_i \in \{C_i - R\}\}$, where X_i is a point of cluster C_i which is closest to q, $d(q, X)$ is the distance between q and X.

(b) $Wi \leftarrow Wi + 1$.

(2) The query contraction motion q is classified by cluster m, where W_m is the maximum value of vector W.

Figure 9: Collecting the sensor data of pelvic floor muscle contraction.

4. Experimental Results

Rehabilitation procedures are divided into four phases. In the first phase, the patient, with the assistance of a doctor, learns how to use the PFMT device, which is a specialized aid used for urinary incontinence rehabilitation. The PFMT device is combined with an embedded system device to detect rehabilitation movement. This assistive device is placed between thighs to measure the force of pelvic floor muscle contractions. In this phase, the sensor data of the patient for pelvic floor muscle contraction is collected at the hospital, as shown in Figure 9.

In the second phase, the k-means algorithm is used to classify collected sensor data collected during the first phase. The data classifications will be used as training data, so that the doctor can examine a patient's condition and prescribe personalized treatment. Doctors, according to classification results, establish guidelines using the characteristics of each cluster corresponding to the patient's condition.

Figure 9 shows the classification results of a PFMT exercise by k-means clustering, in which $k = 4$. The doctor, according to classification results, can prescribe personalized treatment. Figure 10 shows that cluster 1 is the data of patients with correct pelvic floor muscle contraction; the data in cluster 3 and cluster 4 is incorrect contraction motion. The contraction time is shorter than normal contractions in cluster 2 and cluster 3.

In the third phase, sensor data of patient rehabilitation at home is collected. The proposed system uses k-NN algorithm to classify the sensor data and determine whether the patient's contraction actions are correct, according to medical treatment and guidance feedback.

In Figure 11, for example, when the patient is performing pelvic floor muscle contractions, the quantified results, such as *P1* and *P2*, use the k-NN algorithm to find the nearest neighbors of *P1* and *P2*. Of *P1*'s seven nearest neighbors, there are four neighbors in the cluster 4, two in cluster 3, and only one neighbor in cluster 1. Therefore, *P1* is classified as cluster 4; it is an incorrect contraction action. Thus, the proposed system will send a warning message to the patient. Of *P2*'s seven nearest neighbors, four are in cluster 1, and three are located in cluster 4; therefore, *P2* is classified as belonging to cluster 1 and is a correct pelvic floor muscle contraction.

During the last phase, the doctor can follow a patient's rehabilitation progress through charts, in terms such as whether the patient has committed to daily rehabilitation, whether performed pelvic floor muscle contractions are correct or not, and so forth. A rehabilitation data in 15 days is

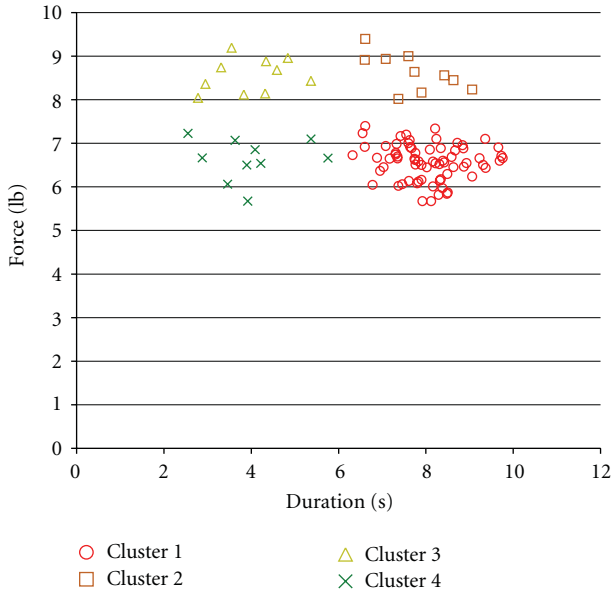


Figure 10: Classification results of a PFMT exercise by k-means clustering, $k = 4$.

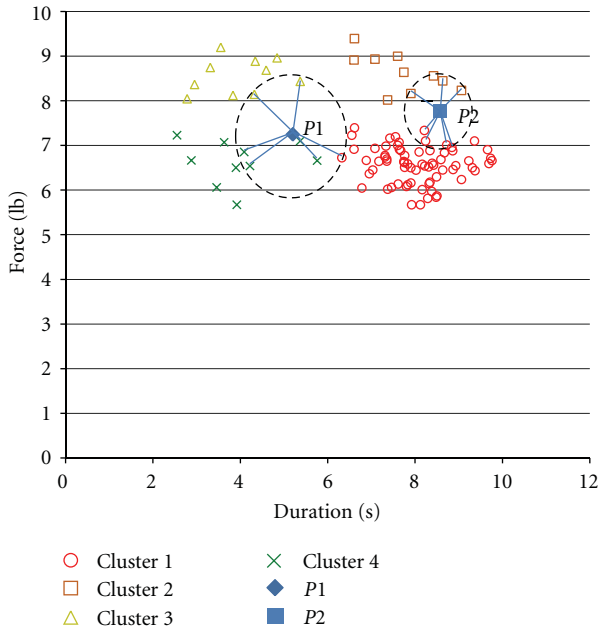


Figure 11: The classification of two pelvic floor muscle contractions *P1* and *P2*.

shown in Figure 12. When the patient returns to the hospital, the doctor can adjust the course of rehabilitation to help patients accelerate recovery.

5. Conclusions and Future Work

Because of privacy issues, many patients reject the invasive treatment of urinary incontinence treatment. Therefore, patient rehabilitation is quite slow in progress [23].

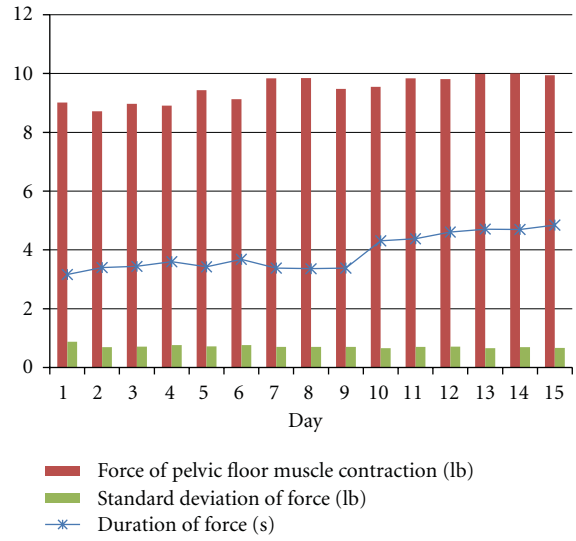


Figure 12: A rehabilitation data in 15 days.

In this study, we use wireless sensors to develop a noninvasive PFMT device for use in a treatment system. This study has two main objectives. First, the patient can do PFMT exercise, at home and without the assistance of doctors or nurses, by using the proposed system; the system can analyze patient rehabilitation sensor data in real time, and patients can determine whether their rehabilitation actions are correct or not and adjust their own rehabilitation actions accordingly to improve the effectiveness of rehabilitation.

Secondly, to deliver better medical care to patients, the system can assist doctors in collecting sensor data, understanding the patient's rehabilitation process, adjusting the course of treatment, and helping patients to improve rehabilitation exercise.

Currently, we have made 20 sets of devices to provide the doctors of the Taipei Veterans General Hospital, Taiwan, for clinical trials. In the future, we will continue to collect patients' feedback to improve our system. Moreover, we will develop new devices and systems to help patients who are ill-suited for standing do PFMT exercise. From a longer-term point of view, we desire to build a more intelligent system that patients can handle by themselves. It will improve the utility of home rehabilitation.

References

[1] S. Zürcher, S. Saxer, and R. Schwendimann, "Urinary incontinence in hospitalised elderly patients: do nurses recognise and manage the problem?" *Nursing Research and Practice*, vol. 2011, Article ID 671302, 5 pages, 2011.

[2] J. C. Lukban, "Transurethral radiofrequency collagen denaturation for treatment of female stress urinary incontinence: a review of the literature and clinical recommendations," *Obstetrics and Gynecology International*, vol. 2012, Article ID 384234, 6 pages, 2012.

[3] G. W. Davila, "Nonsurgical outpatient therapies for the management of female stress urinary incontinence: long-term

effectiveness and durability," *Advances in Urology*, vol. 2011, Article ID 176498, 14 pages, 2011.

[4] K. F. Hunter, C. M. Glazener, and K. N. Moore, "Conservative management for postprostatectomy urinary incontinence," *Cochrane Database of Systematic Reviews*, vol. 2, no. 2, 2007.

[5] M. Kashanian, S. S. Ali, M. Nazemi, and S. Bahasadri, "Evaluation of the effect of pelvic floor muscle training (PFMT or Kegel exercise) and assisted pelvic floor muscle training (APFMT) by a resistance device (Kegelmaster device) on the urinary incontinence in women "comparison between them: a randomized trial"," *European Journal of Obstetrics Gynecology and Reproductive Biology*, vol. 159, no. 1, pp. 218–223, 2011.

[6] P. B. Neumann, K. A. Grimmer, and Y. Deenadayalan, "Pelvic floor muscle training and adjunctive therapies for the treatment of stress urinary incontinence in women: a systematic review," *BMC Women's Health*, vol. 6, article 11, 2006.

[7] E. Kovoor, S. Datta, and A. Patel, "Pelvic floor muscle training in combination with another therapy compared with the other therapy alone for urinary incontinence in women," *Cochrane Database of Systematic Reviews*, vol. 2, Article ID CD007172, 2008.

[8] J. Hay-Smith, S. Morkved, K. A. Fairbrother, and G. P. Herbison, "Pelvic floor muscle training for prevention and treatment of urinary and faecal incontinence in antenatal and postnatal women," *Cochrane Database of Systematic Reviews*, no. 1, Article ID CD007471, 2009.

[9] R. MacDonald, H. A. Fink, C. Huckabay, M. Monga, and T. J. Wilt, "Pelvic floor muscle training to improve urinary incontinence after radical prostatectomy: a systematic review of effectiveness," *British Journal of Urology International*, vol. 100, no. 1, pp. 76–81, 2007.

[10] A. Patel, S. Datta, and E. T. Kovoor, "Pelvic floor muscle training versus other active treatments for urinary incontinence in women," *Cochrane Database of Systematic Reviews*, no. 2, Article ID CD007173, 2008.

[11] C. Dumoulin and J. Hay-Smith, "Pelvic floor muscle training versus no treatment for urinary incontinence in women. a cochrane systematic review," *European Journal of Physical and Rehabilitation Medicine*, vol. 44, no. 1, pp. 47–63, 2008.

[12] A. Marques, L. Stothers, and A. Macnab, "The status of pelvic floor muscle training for women," *Journal of the Canadian Urological Association*, vol. 4, no. 6, pp. 419–424, 2010.

[13] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient *k*-means clustering algorithms: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 881–892, 2002.

[14] H. Xiong, J. Wu, and J. Chen, "*K*-means clustering versus validation measures: a data-distribution perspective," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 39, no. 2, pp. 318–331, 2009.

[15] A. Vattani, "*K*-means requires exponentially many iterations even in the plane," in *Proceedings of the 25th Symposium on Computational Geometry (SCG '09)*, pp. 324–332, June 2009.

[16] T. Tarpey, "Linear transformations and the *k*-means clustering algorithm: applications to clustering curves," *Journal of NIH Public Access*, vol. 61, no. 1, pp. 34–40, 2007.

[17] M. M. Masud, J. Gao, L. Khan, J. Han, and B. M. Thuraisingham, "Classification and novel class detection in concept-drifting data streams under time constraints," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 6, pp. 859–874, 2011.

[18] S. Sun and R. Huang, "An adaptive *k*-nearest neighbor algorithm," in *Proceedings of the 7th International Conference on Fuzzy Systems and Knowledge Discovery, (FSKD '10)*, vol. 1, pp. 91–94, August 2010.

[19] M. Hua, M. K. Lau, J. Pei, and K. Wu, "Continuous *K*-means monitoring with low reporting cost in sensor networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 12, pp. 1679–1691, 2009.

[20] Y. Hong and S. Kwong, "Learning assignment order of instances for the constrained *K*-means clustering algorithm," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 39, no. 2, pp. 568–574, 2009.

[21] B. X. Liu and X. Cheng, "An incremental algorithm of support vector machine based on distance ratio and k nearest neighbor," in *Proceedings of the IEEE International Conference on Computer Science and Automation Engineering (CSAE '01)*, vol. 1, pp. 18–20, June 2011.

[22] K. H. Ambert and A. M. Cohen, "K-information gain scaled nearest neighbors: a novel approach to classifying protein-protein interaction-related documents," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 1, pp. 305–310, 2012.

[23] E. Dugan, C. P. Roberts, S. J. Cohen et al., "Why older community-dwelling adults do not discuss urinary incontinence with their primary care physicians," *Journal of the American Geriatrics Society*, vol. 49, no. 4, pp. 462–465, 2001.

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 764186, 9 pages
doi:10.1155/2012/764186

Research Article

Design and Implementation of a Cyber Physical System for Building Smart Living Spaces

Zhi-yong Bai and Xin-yuan Huang

School of Information Science and Technology, Beijing Forestry University, 35 Tsinghua East Road, Beijing 100083, China

Correspondence should be addressed to Zhi-yong Bai, baizhiyong@vip.qq.com

Received 14 December 2011; Accepted 15 March 2012

Academic Editor: Chih-Yung Chang

Recently, the integrated control systems with computing, Internet, and the electronic equipment with embedded sensors or identification device, called Cyber Physical Systems (CPSs), have gradually attracted more and more attention in a variety of different areas. It is efficient to manage, control, monitor, and query on machinery, equipment, and personnel state. Based on the CPS technology, creating smart living space becomes an important trend of future development. However, the electronic devices nowadays execute different communication protocols, such as Bluetooth, Zigbee, RF, and infrared, and even some traditional devices have no communication function. As a result, the real-time information and status of devices in smart spaces could not be effectively integrated, and thus will increase the difficulties to establish smart living space with CPSs. To improve this problem, this paper designs and implements an Intelligent Control Box to convert different wireless signals. The developed Intelligent Control Box can be treated as a multiple control platform which integrates the systems of lighting, air conditioning, access control, video surveillance, alarm, and so on, further decreasing the difficulties in establishing smart living space with CPSs.

1. Introduction

The Cyber Physical System (CPS) technology has been gradually matured and attracted more and more attention from all over the world. Its related applications have a huge impact in term of human life. In 2007 AD, the President's Council of Advisors on Science and Technology reported eight key technologies which could enhance the modern life, and CPS technology was the primacy. Consequently, the development of CPS technology will be the main trend in the future.

Nowadays, many CPS applications have been proposed, which are *smart space*, *healthcare*, *smart transportation,* and so on. In smart space application, people can use smart phone or tablet to control the daily appliances via remote Internet access, hence increasing the quality of life. Healthcare application can help doctors observe the vital signs of the patients (or the elders) even if the patients (or the elders) stay at home instead of at hospital. In smart transportation application, sensor nodes can be embedded in vehicles to detect the nearby environmental information. For example, the accelerometer and GPS receiver can be embedded in a vehicle. When the accelerometer detects the pothole on the road, the vehicle can send the current coordinates which is obtained by the GPS receiver to the nearby vehicles and thus the traffic safety and efficiency can be improved.

The CPS mainly integrates computing with hardware control. Not only it can enhance the efficiency of system operation, but also it has the ability of monitoring and electronic devices control [1, 2]. Unlike the traditional embedded systems, the CPS is mainly designed for connecting physical devices to build an interaction network. The common way to build CPS is to embed sensors and actuators into electronic devices in daily life. The information of environment and electronic devices usage collected by sensors will be sent to the Decision Making System or the user by the existing Wireless Sensor Network (WSN) techniques, such as routing, data gathering, and MAC protocols [3–7]. Upon receiving the information, the Decision Making System or the user analyzes the collected information and then reflects the decision to the actuators by a sequence of control processes, controlling the electronic devices to perform the corresponding task.

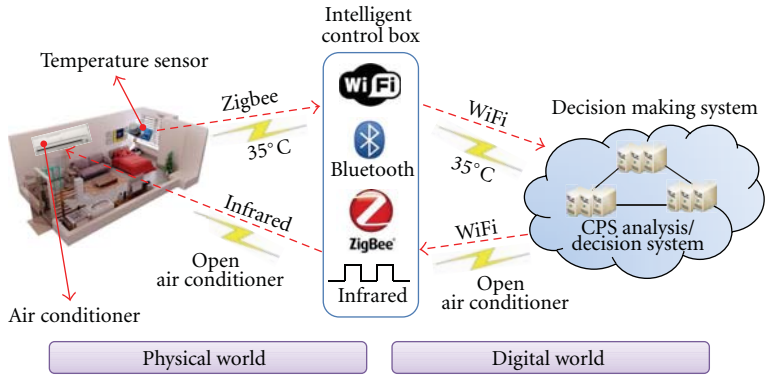


Figure 1: The developed Intelligent Control Box can efficiently integrate different devices even if they execute different communication protocols, further decreasing the difficulties in building smart living space with CPSs.

The CPS is efficient to manage, control, monitor, and query on machinery, equipment, and personnel state and will be the important trend of future development to create more smart living spaces with CPSs technology. However, the electronic devices execute different communication protocols, such as Bluetooth, Zigbee, RF, and infrared. Indeed, some traditional devices have no communication function. As a result, the real-time information and status of devices in smart spaces could not be effectively integrated, thus will increase the difficulties in establishing smart living space with CPSs. To improve this problem, this paper designs and implements an Intelligent Control Box to convert different wireless signals. The developed Intelligent Control Box can be treated as a multiple control platform which integrates the systems of lighting, air conditioning, access control, video surveillance, alarm, and so on, further decreasing the difficulties in establishing smart living space with CPSs.

For example, as shown in Figure 1, if the temperature sensor detects that the room temperature is too high, it will send the "temperature too high" signals to the Decision Making System. However, the Decision Making System may only support Wi-Fi communication protocol. Hence, to successfully send the message to Decision Making System, the temperature signals have to be converted from Zigbee signals to Wi-Fi signals through the developed Intelligent Control Box. After that, the Decision Making System analyzes the information and then sends "turn on the air conditioner" via Wi-Fi communication protocol. Since most air conditioners available nowadays only support infrared technology, the signals must be converted from Wi-Fi signals to infrared signals through the Intelligent Control Box again. As a result, the air conditioner could successfully receive instructions and then start. This example shows that the developed Intelligent Control Box can efficiently integrate different electronic devices even if they execute communication protocols, further decreasing the difficulties in establishing smart living space with CPSs.

The remaining part of this paper is organized as follows. Section 2 reviews some of well-known applications of CPSs while Section 3 introduces the hardware platforms and firmware development environments used in the developed Intelligent Control Box. Section 4 details the key techniques in implementing Intelligent Control Box. Finally, the performance of Intelligent Control Box and conclusions of this paper are drawn in Sections 5 and 6, respectively.

2. Related Work

With development of wireless communication and embedded technology, as well as mature of short-range wireless communication technology, the CPS is primarily used in many domains, including healthcare [8, 9], emergent system [10, 11], transportation [12–15], and smart space [16–23].

In studies [8, 9], people equip with several wearable sensors to acquire the vital signs. Once the wearable sensor detects a specific event, it will send a message to the Decision Making System via Internet. Upon receiving the message, the system offers the corresponding service to the people. Studies [10, 11] developed the emergency system, which guided people to the safe areas when any dangerous event suddenly occurs. References [12–15] designed intelligent transportation systems which make people safer and more convenient.

The Distributed Robotic Garden [16] is a set of garden automated management system with CPS technology developed by MIT. In this system, each plant is equipped with both temperature and humidity sensors to monitor the current states of the garden plants. When a sensor detects the soil moisture supply is low, it sends the current plant location and related information to the Decision Making System. After that, the Decision Making System analyzes the received information and makes a watering decision to the robot in the garden.

The CPS can also be applied to the applications of energy conservation and smart living space. In energy conservation application [17–19], the proposed system can integrate the air conditioner, lighting, elevators, and other electrical equipment on each floor to achieve the energy conservation purpose. For example, when the temperature sensor detects that the floor temperature is too high, it sends this message to the Decision Making System. Upon receiving the message, the Decision Making System analyzes the received information and then makes the decision to "lower the temperature." Contrarily, when the temperature is too low, it makes the decision to "increase the temperature." Based on this system,

the energy efficiency can be dramatically improved, saving unnecessary power consumption.

In application of smart living space, reference [20] developed intelligent devices which are suitable for smart space application. In addition, many technology companies have raised home appliance control system for integration of a variety of common electronic products in the daily life, such as Medusa, Google, and Samsung. Among them, Medusa developed a home appliance control platform and used a peer-to-peer architecture to control the networked multimedia devices [21].

In the Google I/O 2011 conference, Google released Android@Home concept [22], which extended the Android from a mobile or tablet PS device to the household appliances to build smart living space. Android@Home developed a signal converter box. When users intend to control appliances, they can send signals through the Android device to the signal converter box. Upon receiving the signal, the box then sends a corresponding signal to control the appliances.

South Korea's Samsung Company had also launched an intelligent home control system, namely, Homevita [23]. This system mainly integrates Internet and home networks to monitor and control various home electronics devices using remote connection technology.

However, each electronic equipment used in [21–23] shall be developed by the manufacturing company. Otherwise, the real-time information and device state cannot be effectively integrated, and thus increase the difficulty to build smart living space.

According to the aforementioned applications, we can clearly find that the CPS technology plays an important role in creating smart living space. Nonetheless, how to effectively integrate the devices with different communication standards is a big challenge. To this end, this paper designs and implements an Intelligent Control Box to convert different wireless signals. The developed Intelligent Control Box can be treated as a multiple control platform which integrates the systems of lighting, air conditioner, access control, video surveillance, alarm, and so forth, further decreasing the difficulties in establishing smart living space with CPSs.

3. System Hardware and Software

This section firstly introduces the hardware platforms used in the developed Intelligent Control Box and their firmware development environments. Then a detailed description of the software development will be followed.

3.1. Hardware Platform. This subsection aims to introduce the hardware platforms used in Intelligent Control Box, including Octopus II sensor node, Arduino embedded controller board, and the Intel 8051 microcontroller.

3.1.1. Octopus II Sensor Node. Octopus II sensor node is jointly developed by the teams of Taiwan National Tsinghua University and Taiwan National Central University. It adopts a wireless communication module, CC2420 RF chip module, and the Zigbee protocol [24]. Compared with Tmote Sky

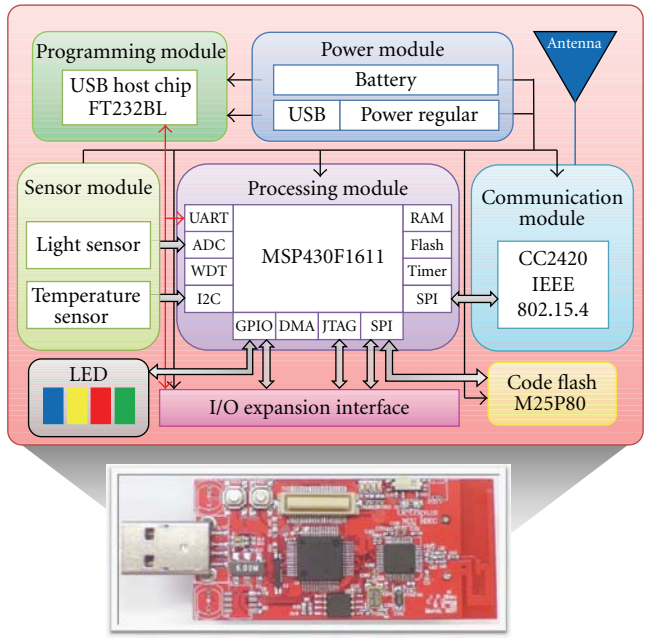


Figure 2: The hardware system architecture of Octopus II sensor node.

sensor node produced by Moteiv Company, Octopus II sensor node is more powerful in terms of computing and data storage space. Furthermore, it can be burned directly with computer programs to facilitate program development and integration. Octopus II adopts dual-antenna design, and its transmission distance can reach to 200 meters. Figure 2 shows the hardware system architecture of Octopus II sensor node.

The Octopus II hardware system architecture mainly consists of Power Module, Programming Module, Sensor Module, Processing Module, and Communication Module. Power Module is mainly responsible for the supply of power required for sensor nodes. The power resource of each node can be provided by two AA batteries; Programming Module is responsible for burning part of the program through the UART of computer interface. Sensor Module is responsible for the collection and analysis of environmental information and converts information from the analog signal to digital signal. Then, the digital information will be sent to the Processing Module. Processing Module is responsible for the operation and control of sensor nodes. This module contains the MP25P80 chip produced by ST Company and MSP430F1611 microprocessor, where MP25P80 chip is used to store program code. When the MSP430F1611 microprocessor is on the operation, the corresponding program code will be selected from the MP25P80 chip. The Communication Module which contains CC2420 chip is responsible for data collection and delivery via wireless transmission technology.

3.1.2. Arduino Embedded Controller Board. Arduino embedded controller board is jointly designed by Professors David Cuartielles and Massimo Banzi. It is a control panel based

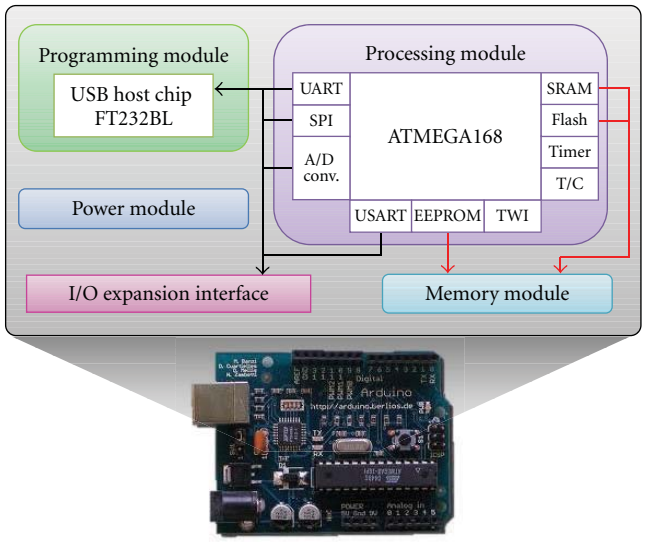


Figure 3: The Arduino hardware system architecture.

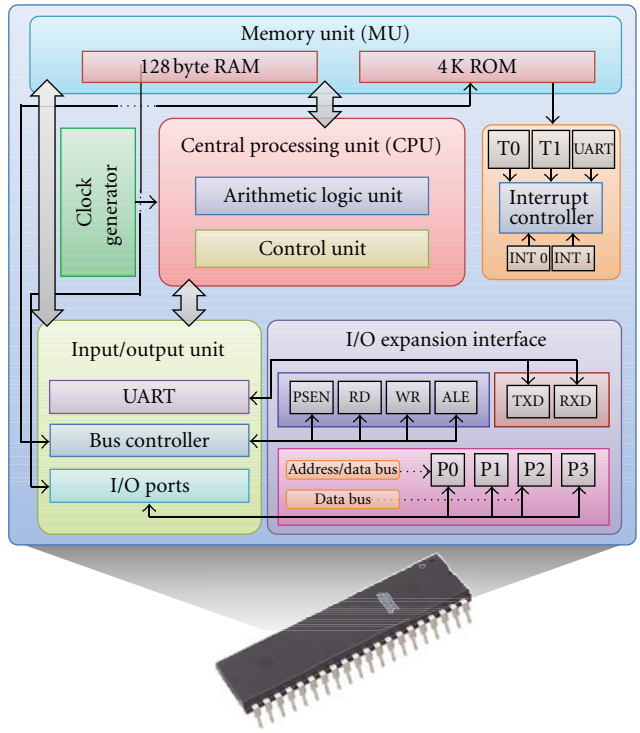


Figure 4: The hardware system architecture of Intel 8051 microcontroller.

on open source developed by the I/O interface with the language similar to Java or C. Ardiuno control panel supplies 14 Digital I/O and 6 Analog I/O and supports USB data transfer. Through the digital outputs, users can connect different electronic devices, such as LED lights, speakers, and motors, and then control them by Controller. In addition, it also can be integrated with Flash or Processing to make real human-computer interactive works. The following details the Arduino hardware system architecture.

As shown in Figure 3, Arduino hardware system architecture mainly consists of Power Module, Programming Module, Memory Module, and Processing Module. Power Module is responsible for the power supply, and the power source can be connected via USB interface or using the 5 V to 9 V DC power supply; Programming Module is responsible for the burning part of the program through the computer's UART communication interface; Memory Module is responsible for managing the EEPROM, SRAM, and Flash's memory usage to decrease excessive use of memory and avoid resulting in low work efficiency; Processing Module uses the octet ATMEGA168 Microcontroller and is responsible for data analysis and computing. It also controls the operations of each hardware component and the voltage output of each pin of Arduino control board.

3.1.3. Intel 8051 Single-Chip. Intel 8051 Microcomputer is created by Intel Corporation in 1981. It has several advantages, including small size, being easy to learn, and good scalability. Therefore, it is widely used in various fields.

As shown in Figure 4, Intel 8051 hardware system architecture primarily consists of Central Processing Unit, Memory Unit, and Input/Output Unit. Central Processing Unit is the core of control program, which contains two subunits: Arithmetic Logic Unit and Control Unit. When the Central Processing Unit receives the coming information from Memory Unit or Input/Output Unit, it uses the Arithmetic Logic Unit to perform arithmetic and logical operations, while the Control Unit is responsible for directing and coordination of data transfer between the modules and operation; Memory Unit is responsible for the data storage sent by Input/Output Unit and stores data processed by Central Processing Unit. In addition, Memory Unit, internal program memory (ROM) with 4 K storage space, is expandable up to 64 K, while the internal data memory (RAM) providing 128 Byte storage space is expandable to 64 K; Input/Output Unit is used to send the information to the external Central Processing Unit as an operation, and then the result of operations is outputted to an external device.

The software platform development environment will be further described in the next section.

3.2. Firmware Platform Development Environment. In this section, the firmware platform development environment of the Octopus II sensor node, Arduino embedded controller board, and the Intel 8051 microcontroller will be described. The firmware development environment Octopus II sensor node is TinyOS, and the compile environment of Arduino embedded control panel is provided by the development Arduino team. The development environment of the Intel 8051 microcontroller is Keil C51.

3.2.1. TinyOS. TinyOS is an embedded operating system [25] designed for wireless sensor nodes developed, and it uses nesC programming language to develop the firmware. In order to improve inherent hardware problems of the sensor node's low memory capacity and MCU slow operation, low flash to burn many programs and limited battery power

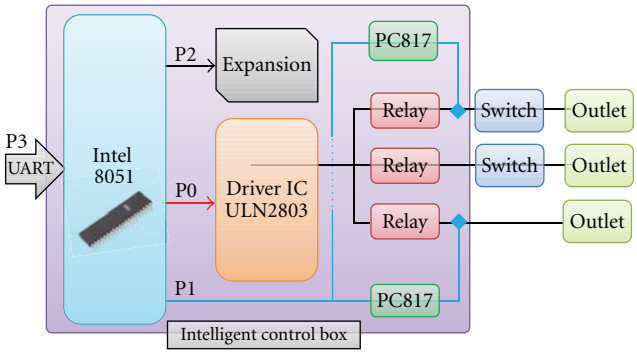


Figure 5: Hardware architecture chart with Intel 8051 microcontroller.

limitations, TinyOS has many frequently used functions modular in advance, such as LED control program, RF transmission control program, and control program of the sensing element, so that users do not need to write too much code to save the sensor node within the limited memory space.

3.2.2. Arduino Firmware Compiler Environment. It has a large library and offers a variety of pin single-chip ATMega information for developers to use. Because the Arduino is an open source platform, developers can write their own libraries. When the error occurs after compiling, the error will be marked immediately to prompt developers. In addition, Ardunio firmware build-in environment with the function of monitoring serial ports can monitor whether message packet is correct through this interface.

3.2.3. Keil C51. Keil C51 is 51 series single-chip firmware integrated development interface developed by Keil Software Corporation in the United States, with C programming language to develop firmware, and can be used to develop Intel 8051 microcontroller firmware in Windows development environment. Its software provides a rich library and powerful debugging tools, and code editor is also available to develop the firmware. It provides a number of single-chip drivers, and C language, assembly language of system components for developers. In addition, it also provides simulator to allow users to check their code. If the code has no error, users can burn the program to the 8051 microcontroller, dramatically saving much time. In the user interface part, it is similar with Microsoft Visual C++ interface. Even if users use the Keil C51 first time, they can easily create a variety of single-chip firmware.

4. The Software and Hardware Development of the Multiple Embedded Home Appliance Control Box

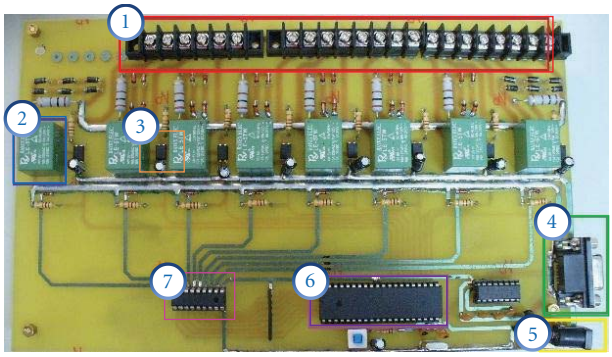
In this section, we will detail the development processes including two parts: hardware architecture and firmware programming. In hardware architecture design, we will use the Octopus II sensor node, Arduino embedded controller board, and Intel 8051 microcontroller to implement the hardware of the developed Intelligent Control Box. In firmware architecture design, we will use TinyOS, Arduino firmware compiler, and Keil C51 to develop the firmware part of Intelligent Control Box.

4.1. Hardware Architecture Design. In this subsection, the implement of hardware is divided into two phases, Control Box Implementation (CBI) Phase and Enhanced CBI (E-CBI) Phase. Intel 8051 microcontroller is used in CBI Phase, while Arduino embedded controller board and Octopus II sensor node are used to connect to the Internet in E-CBI Phase to make user control Intelligent Control Box via remote connection.

4.1.1. Control Box Implementation (CBI) Phase. Shown in Figure 5, the Intel 8051 microcontroller is the main part of the control box. Through UART communication interface, the Decision Making System sends commands. The control box receives and follows the instructions to switch the relay (Relay) of the switch to turn on circuit to make installation able to operate. However, the Intel 8051 microcontroller cannot suffer from the higher voltage and current. To cope with this problem, additional IC ULN2803 chip is added on the Intel 8051 microcontroller. Based on this chip, Intel 8051 microcontroller can successfully control the relay control panel and will not cause excessive current chip damage. In order to detect whether the circuit is turned on, we also add PC817 photo coupler on Intel8051 control panel. The implementation approach is to put two power supply circuits paralleled with PC817 photo coupler and use Intel 8051's Port1 as a detection pin. The internal structure of PC817 photo coupler is an LED and a phototransistor. When the input LED receives a current through it, it will light and the light will turn output transistor on. The potential change is to detect whether the circuit is on.

The circuit inside Intel 8051 single-chip controller board is shown in Figure 6. It is through Printed Circuit Board (PCB) Layout, and welded with relays, resistors, diodes, and other types of IC after circuit washing out. In the chart, the part with the red border is set aside for connection with the outlet or double-cut switch; the part with the blue border is Relay and can open or close the corresponding switch according to the signals sent by Intel 8051; the part with the orange border is PC817 photo coupler and used to detect potential changes and prompt the user whether the circuit is in conduction; the part with the green border is RS232 communication interface to communicate with the computer and control; the part with the yellow border is the power input of the control panel and can be connected to 5 V DC power supply; the part with the pink border is IC ULN2803 and used to withstand high-voltage and high-current to avoid burning of Intel 8051; the part with the purple border is Intel 8051 Clip to analyze signals.

4.1.2. Enhanced CBI (E-CBI) Phase. In the E-CBI Phase, we use the Arduino controller board, Octopus II sensor nodes, Intel 8051 single-chip control panel, WiFi module, Bluetooth module, and infrared modules, to make the Intelligent



❶ Expansion device port
❷ Relay
❸ PC817 photo coupler
❹ RS232 communication interface
❺ Power supply
❻ Intel 8051
❼ Driver IC ULN2803

Figure 6: Internal circuit chart with Inter 8051 microcontroller.

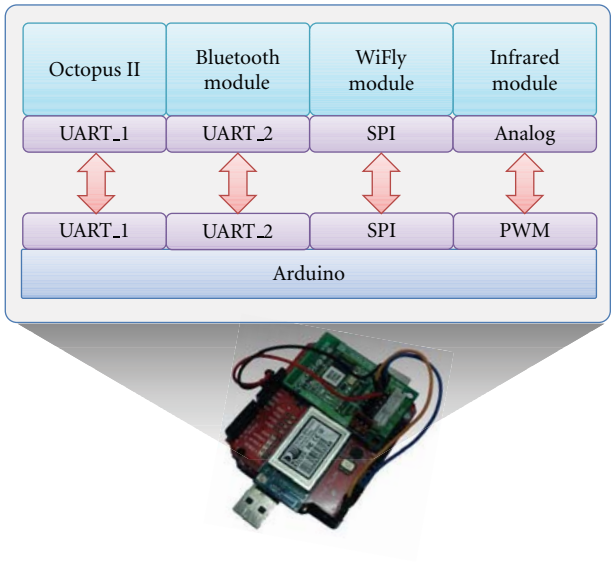


Figure 7: Hardware architecture chart of the developed Intelligence Control Box.

Control Box have capabilities of wireless communication. We focus on this part of the hardware architecture described in detail.

Shown in Figure 7, ZX-BLUETOOTH Bluetooth chip and WiFly GSX 802.11b/g wireless networking chip are integrated into the Arduino control panel. With the control panel of the Intel 8051 single-chip integration, it can connect with a variety of heterogeneous networks. For the Bluetooth chip, it supplies a UART transport interface. The standard operating voltage is 5 V, so we put its UART serial communication interface (5 V, GND, TX, RX) linked on the Arduino control board. After the Arduino UART baud rate (Baud Rate) is adjusted to 9600 bps as Bluetooth chips, it can communicate with the Bluetooth chip. For WiFi chip, the serial communication interface is converted to the SPI through the single-channel high performance SC16IS750 UART chip, and it will enhance the transfer rate between the Arduino and the WiFly GSX.

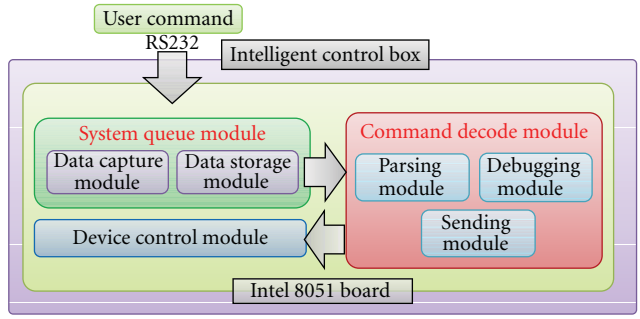


Figure 8: System flow of Intelligence Control Box.

To control electronic devices (such as television and air conditioning) using infrared communication, we also integrate infrared module to the Intelligence Control Box. Infrared transmitter module is integrated with Arduino control board, and Arduino controller board can control the appliances through the PWM output pulse infrared to infrared LED.

4.2. Firmware Development. The implementation of the design of Intelligence Control Box sets aside UART as a communication interface. A variety of devices can use UART and Intel 8051 control panel to connect and control it. If users control or operate them with remote networks, they must have a unified command format. The following is the detailed introduction, how firmware programming is designed to analyze the received packet and format in the Intel 8051 and Arduino and converted into commands to control the box.

4.2.1. Intel 8051 Control Board. In the Intel 8051 control panel, the Decision Making System can also send command packet with the UART interface. To achieve this purpose, a set of algorithms is designed for the program developed by Intel 8051 control panel to correctly parse out the command, and the well-known Longest Common Subsequence (LCS) algorithm is as reference. Its main purpose is to find the longest common subsequence. This algorithm is to find the longest length of the packet format of all set at the beginning in the same time, to facilitate the program to parse packets. We will focus on the detailed description of the system flow chart of the Intel 8051 control panel.

Intel 8051 controller board system operation flow is shown in Figure 8. When the Decision Making System sends packet (User Command) to the Intel 8051 controller board via RS232 communication interface, the packet will go through System Queue Module and then will be temporary stored through the Data Capture Module and Data Storage Module in the Queue way to avoid 8051 microcontroller receiving too many data packets, resulting in system deadlock. When the Intel 8051 microcontroller receives the control packet, it analyzes and parses the packet via Parsing Module in Command Decode Module. After the packets are

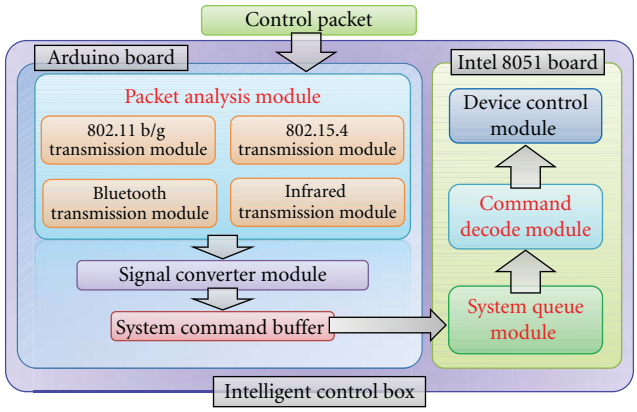


Figure 9: System flow of Intelligence Control Box with Arduino control Board.

parsed and not recognized, information is sent back to computer by Debugging Module and requires the message packet to be resent. Otherwise, control commands are transmitted to Device Control Module by Sending Module for identification to confirm which circuit shall be open or closed.

In this paper, a Buffer is used to save the current command received from the UART and reads and stores the received order with Producer and Consumer algorithm. This algorithm is designed to record current read and stored buffer position with *out* and *in* pointers and performs very smooth under the multithreaded CPU. However, Intel 8051 is a single-threaded CPU. Hence, using this algorithm will produce deadlock. To address this problem, a *Count* pointer is added to record current buffer capacity and avoid deadlock generation. The real way is that each location can be used as a Buffer start position using offset and makes LCS algorithm into a virtual Circle and encountered over all correct buffer points. The program can correctly parse the packet.

The instructions format consists of two formats, the sending instruction and receiving instruction. In sending instruction design, all start byte instructions are 0xFF in transfer instruction format design. The second byte is set to determine whether to read the electrical state connected with the control box or write instructions to control power circuit breaker. The third byte is the second one's extensions, primarily to record every detail action code. The fourth byte is action entrainment data needed by each instruction.

When each instruction is sent to the control box and inquires circuit state or current control circuit in receiving instruction design, the control box will return 10 bytes information to the Decision Making System by default. The state labeled method is that the start bit of the first and second bytes is fixed at 0xAABB, and each followed byte represents the current state of each power supply circuit in use. There are two values 00 and 01 to represent whether power circuit is currently open or closed.

4.2.2. Arduino Control Board. The Arduino Integration Module System Chart is shown in Figure 9. After embedded wireless gateway receives the Control Packet through the wireless network for remote control, it analyzes packets and exchanges information according to users' connection approach. The Packet Analysis Module will automatically switch the networks protocol. After completing the steps above, the data will pass through the Signal Converter Module and be converted into instructions for 8051 single-clip identification which are resent to the control box and temporarily stored in System Command Buffer to confirm whether the queue is full. If there is space available for data storage, which is sent to the System Queue Module, the packet instructions will be analyzed and decrypted through Command Decode Module and finally transferred to the Device Control Module to do allocation for the circuit switching action.

In the firmware writing of the Arduino integrated wireless module, we must write the WiFly GSX 802.11b/g wireless networking chip's control firmware for transferring the data to the Internet via WiFi. In order to develop conveniently, we use WiFly GSX 802.11b/g wireless LAN chip modules specific for Ardino-WiFly Shield to develop. Firstly, we must initialize the SC16IS750 SPI-UART bridge chip on the module through the SPI. We set Baudrate on SPI, transmission format and EFR register, and the serial transfer mode of the TX and RX pin is FIFO. After these settings, WiFly GSX 802.11b/g wireless networking chip can receive instructions and transfer files through Arduino's SPI native function.

In order to make Arduino integrated wireless module having the Zigbee 802.15.4 wireless communications capabilities, we write the serial communication control firmware of the Octopus II to make the Arduino to communicate with the Octopus II using UART. The communication must open the UART mode of MSP430F1611 and then resend control instructions. One way is to directly use the bottom HplMsp430Usart1C components for the UART1 control instructions writing. To use HplMsp430Usart1C to be implemented, you must first declare HplMsp430Usart interface, and then call UsartResource.request() command to request for using the USART mode. TinyOS returns UsartResource.granted() event after request, so UART1 command transmission is well done.

To use Msp430Uart1C components to operate UART1 operation mode, Resource and UartStream interface are being declared. The Resource is responsible for the resource control of UART1, such as resource requirement and resources release. UartStream sends a whole string, and we use an array to hold control instructions.

5. Performance Evaluation

In this section, the performance of the developed Intelligent Control Box is examined. We place Intelligent Control Box at the center position of two smart living spaces whose sizes are 15 m^2 and 30 m^2, respectively.

Let N_t denote the total number of signal conversion during the simulation process and let N_s denote the number of successful signal conversion during the simulation process. Let R_s represent the success rate of signal conversion and it can be defined as $R_s = N_s/N_t$. Figure 10 shows the success rate of signal conversion. As shown in Figure 10, it has

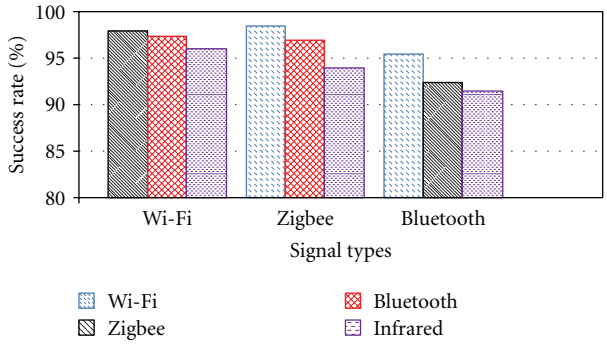


Figure 10: The success rate of converting wireless signals.

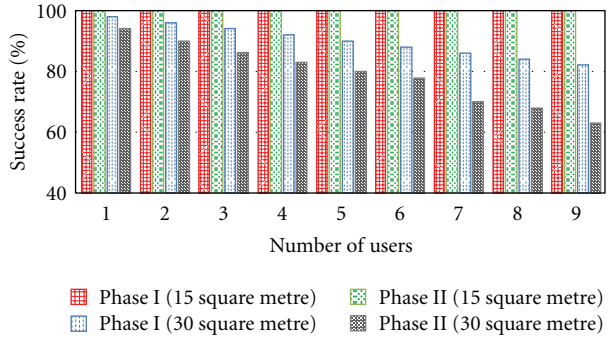


Figure 11: The success rate of controlling electronics in different environment.

high success rates from Zigbee and Bluetooth signals to WiFi signal. The reason is that WiFi signals have the feature of nonline of sight. Similarly, when the WiFi and Zigbee signals are converted into Bluetooth signals, the result is that Bluetooth signals also have the feature of nonline of sight and also have a good signal conversion success rate. Therefore, the infrared signals are in line of sight. Whether WiFi, Zibee or Bluetooth signal is converted to Infrared signal, the signal conversion success rate is relatively low.

In the CPS system, users can also send instructions through User Interface and directly control the various electronic devices. This examination is the comparison of Phase I with Phase II. Phase I is developed for the CBI Phase in the control box with the initial home appliances control in cable transmission, while Phase II is developed for the multiple embedded home appliance control box in this paper. Shown in Figure 11, Phase I uses cable transfers to control electronic equipment, so the success rates of the control electronics both in the environment of 15 m^2 and 30 m^2 are always maintained at 100%. The success rates of Phase II both in the environment of 15 m^2 and 30 m^2 fall along with the decreasing number of electronic devices. This is because the developed control box can only handle a user issued command at one time. The more users here for a time, the lower success rate of electronic equipment control. In addition, the quality of the signal will be weak as the distance increases. Therefore, the performance of Phase II will be lower with the increase of smart living space.

6. Conclusion

In this paper, an Intelligent Control Box is developed to efficiently reduce the difficulties in creating smart living space with CPS technology. The implementation of the control box can be divided into steps, including the CBI Phase and the E-CBI Phase. In CBI Phase, the Intel 8051 microcontroller is used to make a preliminary control box which does not have wireless technology ability to control electronic devices. As a result, the Decision Making System or users only can rely on cable transmission to send commands to the electronic devices. To overcome this problem, the E-CBI Phase is developed in this paper. In E-CBI Phase, with the use of Arduino Control Board, Intel 8051 single chip controller board, Octopus II sensor nodes, WiFi module, Bluetooth module, and infrared modules, the control box is given the abilities of integration with a variety of different communications technologies. Experimental results show that the Intelligent Control Box can convert different wireless signals and decrease the difficulties in establishing smart living space with CPS technology.

Acknowledgment

This work was supported in part by the Beijing Forestry University Young Scientist Fund, under Grant BLX2011020.

References

[1] E. A. Lee, "Cyber physical systems: design challenges," in *Proceedings of the 11th IEEE Symposium on Object-Oriented Real-Time Distributed Computing (ISORC '08)*, pp. 363–369, May 2008.

[2] E. A. Lee, "Cyber-physical systems—are computing foundations adequate?" in *Proceedings of the NSF Workshop on Cyber-Physical Systems: Research Motivation, Techniques and Roadmap*, October 2006.

[3] E. M. Saad, M. H. Awadalla, and R. R. Darwish, "Adaptive energy-aware gathering strategy for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 5, no. 6, pp. 834–849, 2009.

[4] M. Y. S. Uddin, M. M. Akbar, and S. M. Masum, "Hierarchical numbering based addressing and stateless routing scheme for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 5, no. 5, pp. 391–428, 2009.

[5] J. Lian, K. Naik, G. B. Agnew, L. Chen, and M. T. Özsu, "BBS: an energy efficient localized routing scheme for query processing in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2, no. 1, pp. 23–54, 2006.

[6] S. S. Kulkarni and M. Arumugam, "Infuse: a TDMA based data dissemination protocol for sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2, no. 1, pp. 55–78, 2006.

[7] M. L. Pham, D. Kim, S. E. Yoo, and Y. Doh, "Power aware chain routing protocol for data gathering in sensor networks," *International Journal of Distributed Sensor Networks*, vol. 1, no. 2, pp. 253–267, 2005.

[8] Y. M. Huang, M. Y. Hsieh, H. C. Chao, S. H. Hung, and J. H. Park, "Pervasive, secure access to a hierarchical sensor-based

healthcare monitoring architecture in wireless heterogeneous networks," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 4, pp. 400–411, 2009.

[9] G. López, V. Custodio, and J. I. Moreno, "LOBIN: E-textile and wireless-sensor-network-based platform for healthcare monitoring in future hospital environments," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 6, pp. 1446–1458, 2010.

[10] C. Buragohain, D. Agrawal, and S. Suri, "Distributed navigation algorithm for sensor networks," in *Proceedings of the IEEE International Conference on Computer Communications (IEEE INFOCOM '06)*, Spain, April 2006.

[11] M. Li, Y. Liu, J. Wang, and Z. Yang, "Sensor network navigation without locations," in *Proceedings of the IEEE International Conference on Computer Communications (IEEE INFOCOM '09)*, Brazil, April 2009.

[12] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *Proceedings of the ACM Conference on Embedded Network Sensor Systems (ACM SenSys '08)*, November 2008.

[13] A. Thiagarajan, L. Ravindranath, K. LaCurts et al., "VTrack: accurate, energy-aware road traffic delay estimation using mobile phones," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09)*, pp. 85–98, November 2009.

[14] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G. S. Ahn, and A. T. Campbell, "BikeNet: a mobile sensing system for cyclist experience mapping," *ACM Transactions on Sensor Networks*, vol. 6, no. 1, article 6, 2009.

[15] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson, "Cooperative transit tracking using smart-phones," in *Proceedings of the 8th ACM International Conference on Embedded Networked Sensor Systems (SenSys '10)*, pp. 85–98, November 2010.

[16] N. Correll, N. Arechiga, A. Bolger et al., "Building a distributed robot garden," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '09)*, pp. 1509–1516, October 2009.

[17] H. Wicaksono, S. Rogalski, and E. Kusnady, "Knowledge-based intelligent energy management using building automation system," in *Proceedings of the 9th International Power and Energy Conference (IPEC '10)*, pp. 1140–1145, October 2010.

[18] J. Byun and S. Park, "Development of a self-adapting intelligent system for building energy saving and context-aware smart services," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 1, pp. 90–98, 2011.

[19] D. M. Han and J. H. Lim, "Design and implementation of smart home energy management systems based on ZigBee," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1417–1425, 2010.

[20] I. Chun, J. Park, H. Lee, W. Kim, S. Park, and E. Lee, "An agent-based self-adaptation architecture for implementing smart devices in smart space," *Telecommunication Systems*. In press.

[21] S. Wray, T. Glauert, and A. Hopper, "Networked multimedia: the Medusa environment," *IEEE Multimedia*, vol. 1, no. 4, pp. 54–63, 1994.

[22] M. Karch, *Android Tablets Made Simple: For Motorola Xoom, Samsung Galaxy Tab, Asus, Toshiba, and Other Tablets*, Apress, New York, NY, USA, 2011.

[23] Y. Son, S. Ko, J. Jang, H. Lee, J. Jeon, and J. Kim, "Half-push/half-polling," in *Proceedings of the ACM Conference on Pattern Languages of Programs (ACM PLoP '09)*, August 2009.

[24] ZigBee—2006 Specification, ZigBee Document 064112, 2006.

[25] P. Levis and D. Gay, *TinyOS Programming*, Cambridge University Press, New York, NY, USA, 2009.