The Computer Journal Advance Access published June 30, 2014

doi:10.1093/comjnl/bxu054

Resource-Efficient Data Gathering in Sensor Networks for Environment Reconstruction

LINGHE KONG[1], XIAO-YANG LIU[1], MEIXIA TAO[1], MIN-YOU WU[1], YU GU[2], LONG CHENG[3] AND JIANWEI NIU[3,*]

[1]*Shanghai Jiao Tong University, Shanghai, China*
[2]*Singapore University of Technology and Design, Singapore*
[3]*State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University, Beijing, China*
**Corresponding author: niujianwei@buaa.edu.cn*

Environment reconstruction is to rebuild the physical environment in the cyberspace using the sensory data collected by sensor networks, which is a fundamental method for human to understand the physical world in depth. A lot of basic scientific work such as nature discovery and organic evolution heavily relies on the environment reconstruction. However, gathering large amount of environmental data costs huge energy and storage space. The shortage of energy and storage resources has become a major problem in sensor networks for environment reconstruction applications. Motivated by exploiting the inherent feature of environmental data, in this paper, we design a novel data gathering protocol based on compressive sensing theory and time series analysis to further improve the resource efficiency. This protocol adapts the duty cycle and sensing probability of every sensor node according to the dynamic environment, which cannot only guarantee the reconstruction accuracy, but also save energy and storage resources. We implement the proposed protocol on a 51-node testbed and conduct the simulations based on three real datasets from Intel Indoor, GreenOrbs and Ocean Sense projects. Both the experiment and simulation performances demonstrate that our method significantly outperforms the conventional methods in terms of resource efficiency and reconstruction accuracy.

Keywords: wireless sensor networks; compressive sensing; low-duty-cycle; data gathering; environment reconstruction; time series; energy-efficient; data deluge

Received 2 January 2014; revised 6 April 2014
Handling editor: Zhangbing Zhou

1. INTRODUCTION

Environment reconstruction [1] is to rebuild the physical environment in the cyberspace using the sensory data collected by wireless sensor networks (WSNs) [2], which is a fundamental method for human to understand the physical world in depth. A lot of basic scientific work such as nature discovery and organic evolution heavily relies on the environment reconstruction. Several real WSN platforms are deployed under the water [3], in the forest [4] and on the volcano [5, 6] for environment reconstruction applications.

Motivation: It is desired that a WSN can sense and gather the environmental data [7] (such as temperature, light or humidity) with a long-term for accurate environment reconstruction. Nevertheless, this goal is difficult to achieve due to the shortage of resources in WSNs. On one hand, a WSN is constrained by its energy resource. The tiny size of a sensor node and its limited batteries cannot support a long-term data gathering because of the high-energy consumption incurred by wireless communication. For example, a typical TelosB sensor node with CC2420 wireless module consumes $\sim$20 mA for 1-s transmission [8]. But the total power of a TelosB node is no more than 4000 mA capacity [9], which is usually supplied by two AA batteries. On the other hand, a WSN is also constrained by its storage resource. High reconstruction accuracy demands fine-grained sensing, which results in large amount of sensory data and consequently costs huge storage space. To confirm this empirical result, we did an indoor experiment that 51 TelosB nodes gathered over 1 GB data within just 7 days when the sensing interval is set to be 5 s. Furthermore, Baraniuk [10] advocates in SCIENCE that the bottleneck of WSN now is data deluge: the amount of data generated worldwide (1250 billion GB in 2010), which is dominated by sensory data, is growing by 58% per year. For these reasons, energy constraint and data deluge become the key challenges in WSNs, especially for the environment reconstruction application that requires long-term and large-amount data gathering.

Existing approaches and limitation: Existing approaches study either the energy efficiency or the data aggregation problems. (1) Energy constraint is a fundamental problem in WSNs. Hundreds of works have contributed on optimizing the energy consumption from the physical layer up to the application layer [2, 11–14]. In particular, a widely employed approach is to schedule each sensor node's duty cycle (also called wake/sleep cycle) [15–17], which dramatically reduces the energy cost via turning off the radio. Nevertheless, most existing approaches pre-determine the schedule of the duty cycle, which cannot guarantee the reconstruction accuracy for dynamic environment. (2) Data deluge is currently another critical problem in WSNs. Diverse data aggregation approaches have been well investigated for energy balance [18], connected dominating set [19] and energy-latency tradeoff [20]. However, there is no existing work addressing the storage-efficient method with the guarantee of accurate environment reconstruction.

Since both energy and storage efficiency could benefit from reducing the amount of data gathering, they are not incompatible, but correlated mutually in environment reconstruction application. To tackle the joint problem of energy constraint and the data deluge, in this paper, we investigate the resource-efficient data gathering (REDG) problem taking energy efficiency and storage efficiency into account together, which is equal to maximize the sleep duration and to minimize the amount of sensory data gathering, respectively.

Our approach: *First, we analyze the inherent features of environmental data*. After analyzing the real datasets from Intel Indoor experiment [21], GreenOrbs project [4] and Ocean Sense project [22], we observe that the environmental data exhibit obvious low-rank feature. This observation implies that the data exist high redundancy, so that sensing only a few data are able to rebuild the near-optimal environment according to the advanced compressive sensing (CS) theory [23–28]. Furthermore, we verify that the rank of the environmental data is predictable.

Inspired by the observed features, we propose the novel REDG protocol. REDG periodically computes the optimal parameters, including the duty cycle and the amount of sensory data based on the dynamic environment and then adaptively tunes the data gathering behavior. To compute the optimal parameters, one straightforward approach is to gather all the environmental data and derive them in a posteriori manner. However, a posteriori manner might not be feasible because the optimal parameters need to be pre-determined to schedule the data gathering. To tackle this challenge, our design takes advantage of the rank prediction. More specifically, given a requirement of the reconstruction accuracy, by exploiting the predicted rank, we derive the maximum sleep duration using time series analysis [29] based on the predictable rank feature. Meanwhile, we derive the least amount of sensory data using CS theory [30] based on the low-rank feature. Therefore, REDG can effectively adapt to the dynamic environment and achieve REDG.

Finally, REDG is implemented on a testbed and simulated on three real datasets. We evaluate the proposed REDG, the standard collection tree protocol (CTP) [31] and the classic low-duty-cycle protocol [15] on a 51-node real testbed for performance comparison. A 30-day experiment shows that REDG significantly outperforms the other protocols on energy and storage efficiencies. To understand the performance of REDG in large-scale sensor networks, we then conduct extensive simulations based on real datasets from Intel Indoor, GreenOrbs and Ocean Sense projects. The evaluation results also demonstrate that our REDG achieves low resource consumption with the guarantee of reconstruction accuracy.

Contributions: In summary, the major contributions of this paper are as follows:

(i) To the best of our knowledge, this is the first work to study both the energy and storage efficiency problems in data gathering of WSN for environment reconstruction.
(ii) We mine three real datasets to explore the low-rank and predictable rank features of dynamic environments.
(iii) A novel REDG protocol is designed based on CS theory and time series analysis. This protocol can adapt duty cycle and the amount of data gathering according to the dynamic environment. And the near-optimal accuracy of environment reconstruction is guaranteed.
(iv) The proposed protocol is implemented on a real testbed and simulated based on three real datasets. The results verify its feasibility and show its significant improvement compared with the existing methods.

Paper organization: The remainder of this paper is organized as follows. In Section 2, the background is presented. In Section 3, the problem is formulated. Environment features are mined in Section 4. In Section 5, the novel REDG protocol is proposed. In Section 6, implementation and simulation are performed for evaluating REDG. In Section 7, we conclude this work.

2. BACKGROUND

In this section, we present the concept of environment reconstruction, existing resource-efficient methods and the background of CS theory.

2.1. Environment reconstruction

The objective of environment reconstruction [1] is to rebuild the accurate environment in cyberspace based on the gathered sensory data, which is commonly realized by WSNs. Several real applications can be found in [3, 4, 6]. However, such applications are restricted by the energy and storage resources.

The energy constraint is a classic problem in WSNs. According to [2], battery capacity only doubled in the past 35 years. Moreover, the hazardous sensing environment precludes manual battery replacement. Energy constraint is unlikely to be solved in the near future on account of the size limitation of sensor nodes.

In recent years, the data deluge issue becomes a serious bottleneck of WSNs. A recent report [10] found that the total amount of world data storage is growing 31% slower than the amount of data generated worldwide (dominated by sensor networks). This expanding gap indicates that storage efficiency will be a critical issue in WSNs.

2.2. Existing resource-efficient methods

This work is related to energy-efficient and storage-efficient methods. However, we find that existing approaches cannot satisfy the joint problem of energy constraint and data deluge for environment reconstruction.

In WSNs, the energy-efficient methods are investigated from physical layer [12], link layer [11, 14], network layer [13] to application layer [2]. Particularly, scheduling the duty cycle of every sensor node is the widely employed approach for energy-saving data gathering [15–17, 32]. Although these solutions are highly diverse, none of them considers an adaptive energy-efficient mechanism according to the change of environment.

There are also plenty of data compression and data aggregation approaches, which are studied to reduce the storage consumption in WSNs, e.g. [18–20]. However, we observe that these approaches operate at the sink or relay nodes. In another word, the environmental data actually have been sensed and then be processed. In this case, some storage and energy resources have been used. Hence, in this paper, we study the storage-efficient problem by reducing the amount of data gathering at the source nodes.

2.3. Compressive sensing

CS [23–25] is a generic method to recover the whole condition with only a few sampled data [33–36]. Several effective CS-based applications have been developed in data recovery field. For instance, network traffic estimation [28], road traffic interpolation [26] and localization in mobile networks [27].

CS-based methods have the potential to reconstruct the environment in WSN applications. It has been proved that the environment can be near-optimally recovered even there are more than 70% sensory data are missing [30], which motivates us to exploit CS to reduce the amount of data gathering. Until now, there is no CS-based method having been studied to optimize the data gathering for environment reconstruction.

3. PRELIMINARIES

3.1. System model and notation

In environment reconstruction system, sensor nodes are distributed in the given area to sense and gather data to the sink during a given period of time. Suppose there are totally n sensor nodes. The period of monitoring time is evenly divided into t time slots, any sensor node can periodically (once per time slot) sense the environmental data.

Definition 1. *Environment condition* (EC): EC is the real environmental data sensed by sensor node i at the time slot j denoted by $x(i, j)$, where $i = 1, 2 \ldots, n$ and $j = 1, 2 \ldots, t$.

Definition 2. *Environment matrix* (EM): EM is the matrix of all real environmental data, which is a mathematical way to describe the dynamic environment. All ECs $x(i, j)$ form a EM:

$$X = (x(i, j))_{n\times t}. \tag{1}$$

Thereby, this is a matrix constituting of n rows and t columns. A complete EM presents that all ECs are gathered, which indicates the 100% accurate environment reconstruction.

Definition 3. *Binary index matrix* (BIM): BIM is a $n \times t$ matrix, which indicates whether the sensor node i at time slot j senses the EC or not. BIM is defined by

$$B = (b(i, j))_{n\times t} = \begin{cases} 1 & \text{if } x(i, j) \text{ is gathered,} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Definition 4. *Gathered matrix* (GM): GM is the matrix of only gathered data by a WSN. Since some ECs may not be gathered, elements of GM are either EC $(x(i, j))$ gathered by sensor node or zero $(b(i, j) = 0)$. Thereby, GM is an incomplete EM. GM is denoted by G and can be presented by

$$G = X \cdot B. \tag{3}$$

Definition 5. *Estimated environment matrix* (E2M): E2M is the result of environment reconstruction, which is generated by interpolating the missing values in GM to approximate EM. E2M is denoted by

$$\hat{X} = (\hat{x}(i, j))_{n\times t}. \tag{4}$$

3.2. Problem statement

In this paper, we focus on the data gathering problem for saving the resource consumption and acquiring accurate environment reconstruction.

Problem 1. *Resource-efficient environment reconstruction* (REER): Given n sensor nodes, which are scattered in a given area for environment reconstruction during the period of t time slots, the REER problem is formulated by

$$\begin{cases} \text{Objective:} & \min(E), \\ \text{Sub-Objective:} & \min(S), \\ \text{Subject to:} & \|X-\hat{X}\|_F \leq \xi, \end{cases} \tag{5}$$

where E is the energy consumption, S is the storage consumption, ξ is the accuracy requirement and $\|\cdot\|_F$ is the Frobenius norm used to measure the error between matrix X and $\hat{X}$. For describing the calculation of Frobenius norm, we take X as an example that $\|X\|_F = \sqrt{\sum_{i,j}(x(i,j))^2}$.

The REER problem describes that our objectives are to minimize the energy and storage consumptions during the data gathering in a WSN when the accuracy requirement of environment reconstruction is guaranteed.

We decompose and analyze the Problem 1 as follows.

First, this problem subjects to $\|X-\hat{X}\|_F \leq \xi$. To measure the errors of diverse environments by a unified metric, we define the normalized error.

Metric 1. *Normalized environment reconstruction error* (ER-Error):

$$\epsilon = \frac{\sqrt{\sum_{i,j:b(i,j)=0}(x(i,j)-\hat{x}(i,j))^2}}{\sqrt{\sum_{i,j}(x(i,j))^2}}, \tag{6}$$

where the condition $b(i,j)=0$ in Equation (6) presents that only errors at the missing data positions are computed.

The threshold value ϵ_{th} is the normalized accuracy requirement, which can be derived from ξ. If the ER-Error $\epsilon \leq \epsilon_{\text{th}}$, the environment reconstruction is considered to match the accuracy requirement. The ER-Accuracy can be noted as $(1-\epsilon)$.

Secondly, the objective is to $\min(E)$. From [16], we know that the wireless communication costs much more energy than computing and sensing in WSNs. For example, a TelosB node needs only 1.8 mA on computing and sensing, but 19.5 mA on wireless communication [8]. One common energy-saving method is to turn off the radio when the sensor node has no transmission task. Consequently, $\min(E)$ is evolved to maximize the duration of radio off period $\max(t_\theta)$ when the duration of radio on period t_o is given, which is equivalent to minimize the duty cycle.

Thirdly, the secondary optimization objective is to $\min(S)$. The storage consumption S in WSNs is for storing the gathered data. (The maximum value of S is $n \times t$, which presents that all data have been gathered.) Minimizing the storage consumption is a very challenging problem at the source node side. For a sink node, it has all gathered environmental data, so that the least amount key data can be obtained by the analysis method in [1]. However, a source node does not know the global environment, so it cannot make the decision whether to gather the environmental data or not. To address this problem, a novel solution will be introduced in Section 5.

4. ENVIRONMENTAL DATASETS ANALYSIS

In this section, the features of environments are mined before solving the REER problem.

Our analysis is based on three datasets gathered by real WSN projects. The basic information of these datasets are listed in Table 1.

The data of Intel Indoor experiment [21] are gathered by Intel Berkeley Research lab. There are totally 54 Mica2 nodes placed in a 40 m × 30 m room. Every node reports data every 30 s. From all the gathered data, we select 50 nodes × 4000 slots data to form a complete dataset (data loss exists universally in real applications, but we need complete EMs for data mining).

GreenOrbs project [4] is a real WSN application for forest surveillance. More than 450 TelosB nodes are scattered on the Tianmu Mountain, China and gather temperature, light and humidity once every 5 min. We select 249 × 500 environmental data from GreenOrbs.

Ocean Sense project [22] contributes for our third dataset. This dataset contains 20 TelosB nodes deployed in the sea of Taipingjiao, China, which monitors an area of 300 m × 100 m. Each sensing node reports temperature and light data every 2 min. We select 15 × 1000 data from Ocean Sense.

From the selected datasets, we generate six EMs: Indoor-Temp (temperature matrix of the Intel Indoor), Indoor-Light, Forest-Temp, Forest-Light, Ocean-Temp and Ocean-Light for environmental features discovery.

4.1. Low-rank structure discovery

Environmental data at different locations over different times are usually not independent, which may exist low-rank structure, i.e. the data are compressible. To mine the low-rank feature, we analyze the selected EMs from real WSN datasets using principal component analysis (PCA), which is an effective non-parametric technique for revealing low-rank structure [37].

TABLE 1. Real datasets for environmental features analysis

Data name	Matrix size	Slot duration
Intel Indoor	50 nodes × 4000 slots	30 s
GreenOrbs	249 nodes × 500 slots	5 min
Ocean Sense	15 nodes × 1000 slots	2 min

Any $n \times t$ matrix X can be decomposed into three matrices according to singular value decomposition (SVD):

$$X = U \Lambda V^{\mathrm{T}} = \sum_{i=1}^{\min(n,t)} \sigma_i u_i v_i^{\mathrm{T}}, \tag{7}$$

where V^{T} is the transpose of V, U is a $n \times n$ unitary matrix (i.e. $UU^{\mathrm{T}} = U^{\mathrm{T}}U = I_{n\times n}$), V is a $t \times t$ unitary matrix and Λ is a $n \times t$ diagonal matrix constraining the singular values σ_i of X. Typically, the singular values in Λ are sorted. Let $\sigma_i \geq \sigma_{i+1}$, $i = 1, 2, \ldots, \min(n,t)$, where $\min(n,t)$ is the number of σ_i. The rank of a matrix, denoted by r, is equal to the number of its non-zero singular values. If $r \ll \min(n,t)$, the matrix is low-rank. According to PCA, a near low-rank matrix [28] exhibits that its top r singular values occupy the total or near-total sum singular values $\sum_{i=1}^{r} \sigma_i \approx \sum_{i=1}^{\min(n,t)} \sigma_i$.

In Fig. 1, we illustrate the CDF of top-% singular values in the six selected EMs from real datasets. The X-axis presents the number of singular values. Since the sizes of six EMs are different, we normalize the X-axis, where $\min(n,t)$ of each EM is normalized to 100%. The Y-axis presents the cumulative values of top-% singular values. Owing to the same reason, the ordinate is also normalized, where $\sum_{i=1}^{\min(n,t)} \sigma_i$ of each EM is normalized to 100% in Y-axis. This figure implies that the sum of all singular values is always contributed by only a few top singular values in real environments. For example, the top 5% singular values contribute 92% of sum singular values in Indoor-Temp. The universal existence of $\sum_{i=1}^{r} \sigma_i \approx \sum_{i=1}^{\min(n,t)} \sigma_i$ and $r \ll \min(n,t)$ reveals that EMs exhibit obvious low-rank structures. Low-rank features [26] indicate that EMs can be near-optimally rebuilt by CS even if massive environmental data are missing.

4.2. Predictability of rank

Naturally, the value of rank r relies on the dynamic environment. Many works [1, 28] have proved that the change of environment

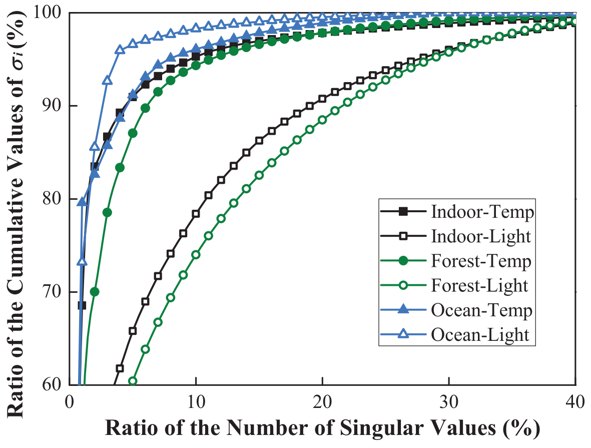


FIGURE 1. CDF of singular values to explore the low-rank feature.

has the feature of adjacent-time stability. This feature motivates us to discover whether the dynamic r is predictable with time series.

The well-known Box–Jenkins model [38] can check whether a time series is predictable while considering its stationarity, intercept term, seasonal factors and exogenous variables. Therefore, we adopt the Box–Jenkins model to identify the predictability of r in the six selected EMs. The identification steps are as follows:

(i) *Rank sequence generation*: Constructing the original rank sequence $R = (r_\tau)_{1\times t}$, i.e. the time series of rank. Since EM X is a $(n \times t)$ matrix, there are totally t time slots. Then, $\tau = 1, 2, \ldots, t$, and $R = (r_1, r_2, \ldots, r_t)$.

We define $X_{n\times\tau}$ is a $(n \times \tau)$ matrix constructed by the first τ column vectors of X. The rank of X_τ is r_τ, which is equal to the number of non-zero singular values after SVD $X_{n\times\tau}$ (refer to Equation (7)). Hence, $R = (r_1, r_2, \ldots, r_t)$ is obtained.

(ii) *Stationarity requirement*: Stationarity is the basic condition of a predictable time series. Therefore, we detect the original rank sequence whether it has stationarity. If it is non-stationary, the difference process is needed to be operated to achieve stationarity.

A sequence (y_τ) is stationary if it satisfies

$$\begin{cases} \exp(y_\tau) = \beta, \\ \operatorname{cov}(y_\tau, y_{\tau+\omega}) = \gamma_\omega, \end{cases} \tag{8}$$

where exp() is the expectation function, β is a constant, cov() is the autocovariance function, ω is the lag coefficient and γ_ω is independent of τ.

If the original rank sequence $R = (r_\tau)$ is not stationary, we take the first-order difference process $\triangledown R = r_\tau - r_{\tau-1}$. If $\triangledown R$ is still not stationary, the second-order difference is calculated $\triangledown^2 R = \triangledown(\triangledown R) = r_\tau - rk_{\tau-1} + r_{\tau-2}$. The d-order differences $\triangledown^d R$ will be computed until the stationary is achieved.

The computation results show that the $\triangledown R$ of Ocean-Temp and Ocean-Light are stationary; and $\triangledown^2 R$ of Indoor-Temp, Indoor-Light, Forest-Temp, Forest-Light suffice the stationarity requirement.

(iii) *Predictability identification*: Once a sequence is stationary, its predictability can be identified by its autocorrelation coefficient figure (ACF) and partial autocorrelation coefficient figure (PACF).

Assume (y_τ) is a stationary sequence, its autocorrelation coefficient $\rho_\omega = \gamma_\omega/\gamma_0$, where γ_ω is obtained based on Equation (8), and $\gamma_0 = \operatorname{cov}(y_\tau, y_\tau)$ is the variance of (y_τ). The partial autocorrelation coefficient φ_ω is obtained by the following recursion formula:

$$\varphi_1 = \rho_1, \tag{9}$$

$$\varphi_\omega = \frac{\rho_\omega - \sum_{l=1}^{\omega-1} \varphi_{(\omega-1,l)}\rho_{\omega-l}}{1 - \sum_{l=1}^{\omega-1} \varphi_{(\omega-1,l)}\rho_l}, \tag{10}$$

where $\varphi_{(\omega,l)} = \varphi_{(\omega-1,l)} - \varphi_\omega \varphi_{(\omega-1,\omega-l)}$. Then, we can plot ACF by ρ_ω and PACF by φ_ω.

Generally, the trend of ρ_ω in ACF and φ_ω in PACF can be divided into two types: either decay to zero exponentially, noted by DECAY; or drop to essentially zero after some spikes, noted by DROP. According to the Box–Jenkins model, the predictability of a sequence can be referred to Table 2, which summarizes the predictability identification by the trend of ρ_ω and φ_ω.

We apply the above three steps on the six EMs. Then, we obtain six ACFs and six PACFs for each selected EM as shown in Fig. 2. Figure 2b exhibits the trend of DECAY and Fig. 2a displays the trend of DROP. We find that for every environmental dataset, there is at least one figure (ACF, or PACF) having the trend of DECAY. For example, the ACFs of Ocean-Temp, Ocean-Light are DECAY; the PACFs of Indoor-Temp, Indoor-Light, Forest-Temp, Forest-Light are DECAY. Thus, the rank sequence R in these EMs has the predictability. The universal results also verify that the dynamic rank, in general, environment is predictable.

5. REDG PROTOCOL

Inspired by the observed features, we design a novel REDG protocol in this section.

5.1. S3 process

Before the overview of REDG design, we first introduce the data gathering process adopted in this work.

TABLE 2. Predictability identification by ρ_ω in ACF and φ_ω in PACF.

ρ_ω in ACF	DECAY	DROP	DECAY	DROP
φ_ω in PACF	DROP	DECAY	DECAY	DROP
Predictable	YES	YES	YES	NO

For a sensor node, there are two kinds of general data gathering processes: *sense-store-send* (S3) and *sense-send* (S2) [8]. Under S3, a node firstly senses the environmental data, then stores the data in the local memory for a period, and finally sends the data to the sink node; while a node with S2 senses the data and sends them to the sink directly.

Different processes lead to different number of operation states for sensor nodes. A node with S2 has two states: active and sleep. In the active state, a node carries out sensing and sending operations when the radio on. In the sleep state, a node turns its radio off and does no operation. On the contrary, there are three states in S3: active, sleep and half-active. During the half-active state, a node carries out the sensing and storing operations while the radio is switched off. The stored data would be kept until the radio is turned on next time.

Our design adopts the S3 process because the half-active state potentially saves lots of energy consumption. Guo *et al.* [16] have investigated that in WSNs, the wireless transmission (no matter sending, receiving or listening) costs much more energies than computation and sensing operations. Hence, the half-active state in S3 can help nodes to save energy through turning the radio off, which meets our energy-saving objective.

The half-active state is easy to be implemented on the off-the-shelf sensor node and the standard sensor OS. Current sensor nodes have the flash memories (e.g. 1 M in TelosB), which can store the sensory data (usually <200 Bytes per record). The radio is an independent module (e.g. CC2420 in TelosB), which can be disabled separately. The current OS (e.g. TinyOS) also offers the function to turn off the radio module.

The S3 process is appropriate to the environment reconstruction applications. Since environment reconstruction always serve for scientific discovery or statistics after all data gathered, such applications do not require the real-time data gathering. Therefore, the data are allowed to be stored in the sensor nodes for a while.

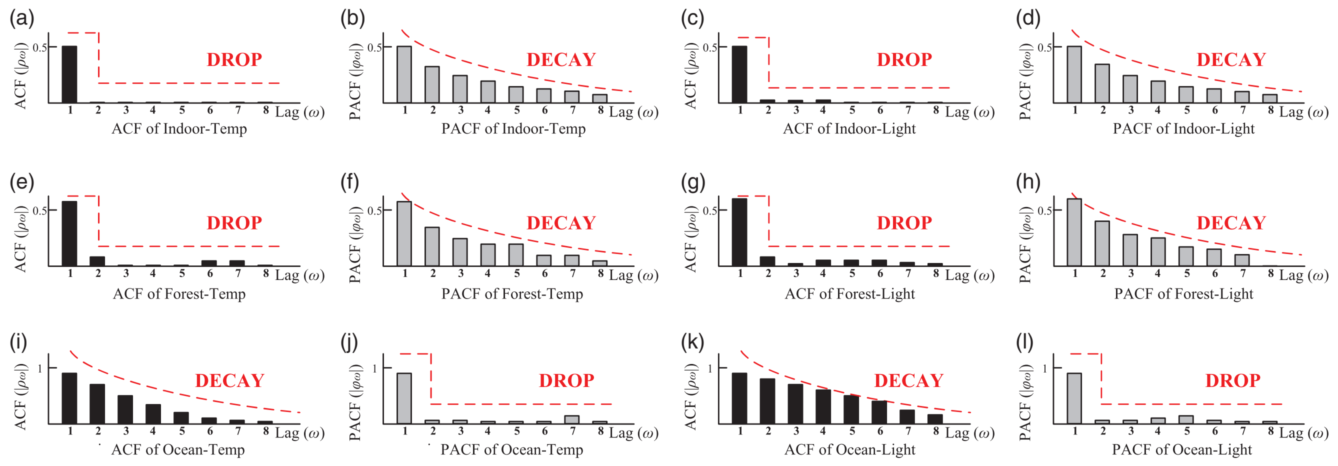


FIGURE 2. ACFs and PACFs of diverse datasets to identify the predictability of rank sequence.

5.2. Design overview

We have mined two features and have adopted one process: (1) The predictable rank feature lays the foundation for the duty-cycle adaption. (2) The low-rank feature indicates that only a small amount of gathered data can recover the whole E2M with high accuracy by CS. This amount of data saves a lot of storage space. (3) The S3 process saves the energy due to half-active state.

Combining the above three advantages, we design the REDG protocol. REDG runs cycle-by-cycle. The procedure of every cycle, including a period of radio on and a period of radio off, is described in detail as follows:

Period of radio on: Nodes in this period are at the active state. The duration of this period is t_o time slots, which is set by the user or by default. First, The sink node gathers all stored data in the last period of radio off from sensor nodes. Secondly, the sensory data in current period from n sensor nodes are sent to the sink as the sample. Thirdly, the sample data are analyzed immediately, so the sensing probability p and the duration of radio off period t_θ are calculated (the calculation method will be introduced in the next two subsections). Fourthly, the sink feedbacks the values of p and t_θ to n sensor nodes. Parameters p and t_θ are valid only in current cycle.

Period of radio off: Nodes in this period includes the sleep and the half-active states. The duration of this period is t_θ time slots obtained by the latest Step V.B.1. At every time slot in this period, each node senses environmental data with the latest sensing probability p, and stores them in local memory until this period is over. Then, the execution goes to 'Period of Radio On' and repeats.

We use an example to compare REDG with the traditional data gathering protocols: CTP and fixed low-duty-cycle (FLDC) as shown in Fig. 3. Assume the number of nodes $n = 3$, and the number of total time slots $t = 30$ for this environment reconstruction application. The black box in Fig. 3 presents the active state; the white box presents the sleep state; and the cross box presents the half-active state. The average duty cycle is equal to the ratio of the number of black boxes to the total number of boxes. The storage consumption is the number of non-white boxes.

We can find that the CTP needs the largest amount of storage space (90 units) and it costs the most energy because the duty cycle is 1. The resource consumption of FLDC, where the duty cycle is 0.4 and the storage space needs only 36 units, is better than CTP. However, FLDC cannot ensure the accuracy by any interpolation method due to no environment adaption. The proposed REDG needs only 30 units storage space. The duty cycle of REDG is only 0.2. In addition, since the parameters of p and t_θ are adapted by the dynamic environment, the reconstruction accuracy can be ensured by these 30 units data using CS method.

We re-analyze the REER problem according to the proposed REDG design.

It is proved that the majority energy consumption is due to radio on, so $\min(E)$ in Equation (5) is equivalent to minimize the average duty cycle of the nodes. Because REDG runs cycle-by-cycle, to minimize the average duty cycle can be realized by minimizing the duty cycle of each cycle. Since t_o is fixed by default in REDG and the computation of duty cycle is $= t_o/(t_o + t_\theta)$, we obtain $\min(E) \Rightarrow \max(t_\theta)$.

The secondary objective $\min(S)$ is equivalent to minimize the storage consumption in each cycle because REDG runs cycle-by-cycle. The storage space in each cycle is $n \times (t_o + t_\theta \times p)$. Since n and t_o are known; t_θ is maximized before; and $p \geq 0$, we obtain $\min(S) \Rightarrow \min(p)$.

In summary, we rewrite Equation (5) to obtain the goal of REDG:

$$\begin{cases} \text{Objective:} & \max(t_\theta), \\ \text{Sub-Objective:} & \min(p), \\ \text{Subject to:} & \epsilon \leq \epsilon_{\text{th}}. \end{cases} \tag{11}$$

5.3. Duty-cycle adaption

We introduce the duty-cycle adaption method in this subsection. According to Equation (11), the objective is $\max(t_\theta)$ for efficient energy consumption. However, t_θ is self-adaptive to

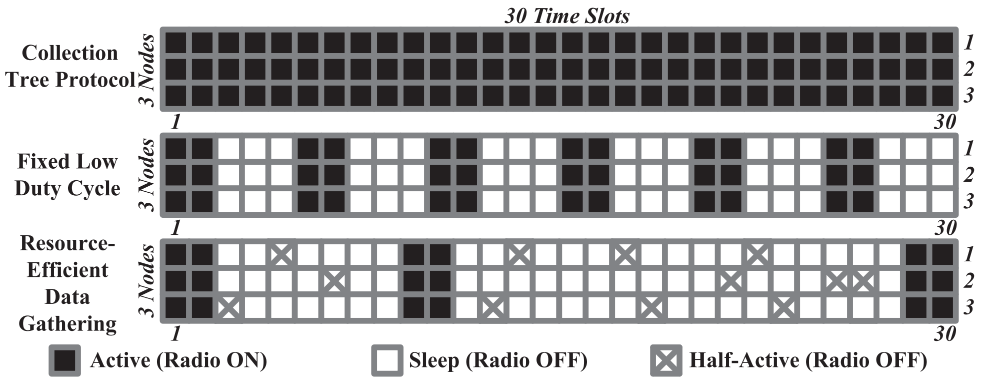


FIGURE 3. Procedure comparison among three data gathering protocols.

the dynamic environment, which is mainly restricted by the accuracy requirement. Consequently, we utilize time series analysis [29] to estimate t_θ on the basis of the sample data during the latest t_o. We decide t_θ by three steps.

(i) *Rank estimation*: We have proved the predictability of ranks in EMs. Here, we introduce the method to estimate the rank sequence. An $n \times t_o$ EM can be obtained from the sample data. Following the Step VI.B.1, a rank sequence $R = (r_\tau)$ is generated, where $i = 1, 2, \ldots, t_o$. ARIMA model [39] in time series analysis can well predict the sequence with the long trend, periodicity and random noise, so we adopt ARIMA to estimate $\hat{r_\tau}$, where $\tau > t_o$.

ARIMA model is a mature prediction tool, which has been widely used in economy prediction and integrated into many statistics softwares such as SAS and SPSS. Therefore, only a brief description of ARIMA process is introduced for $\hat{r_\tau}$ estimation as follows:

(a) Stationarity Difference. Compute $\min(d)$, where $d = 0, 1, 2, \ldots$ and this d satisfies that $\bigtriangledown^d R$ is stationary.
(b) Term Decision. ARIMA model has three terms (a, b, c). The first term is decided by $a = \min(\omega)$, where this ω matches $|\rho_\omega| < 2/\sqrt{\omega}$; The second term is $b = \min(d)$; The third term is $c = \min(\omega)$, where this ω matches $|\varphi_\omega| < 2/\sqrt{\omega}$. Then, the terms (a, b, c) are determined.
(c) Forecast. Use ARIMA(a, b, c) to forecast the value of rank. Then, we get an estimated rank sequence $\hat{R} = (\hat{r_\tau}),\ \tau = t_o + 1, t_o + 2, \ldots, \infty$.

For detailed process the reader can refer to [29].

(ii) *Rank error estimation*: This step is used to predict the error between the real rank and the estimated rank.

In the above step, ARIMA model is employed to predict the value of $\hat{r_\tau}$, where $\tau > t_o$. In this step, we also estimate $\hat{r_\tau}$ but by the sequence $R' = (r_1, r_2, \ldots, r_{\tau-1})$, where $1 < \tau \le t_o$.

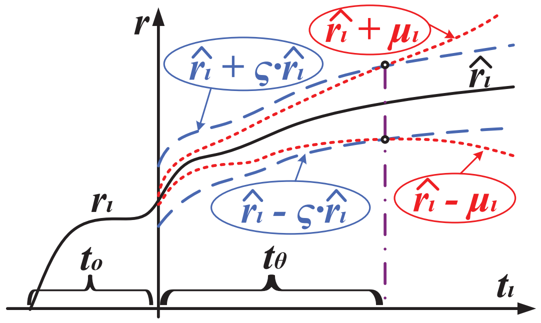
Then, we generate the rank error sequence. Since we have the real r_τ computed directly from the sample data, the rank error $\mathscr{E}$ can be compared between r_τ and $\hat{r_\tau}$ when $1 < \tau \le t_o$. Hence, we can create a rank error sequence $\mathscr{E} = (e_\tau) = (r_\tau - \hat{r_\tau}),\ \tau = 1, 2, \ldots, t_o$.

Finally, the estimated rank error sequence $\widehat{\mathscr{E}} = (\hat{e_\tau})$ can be predicted by the same ARIMA method as mentioned in the step of 'Rank Estimation', where $\tau = t_o + 1, t_o + 2, \ldots, \infty$.

(iii) *Duration decision*: The radio off duration t_θ is determined by the accuracy requirement.

The errors do exist in the estimated rank sequence $\hat{R} = (\hat{r_\tau})$. And the longer the estimated sequence, the larger errors are accumulated. Therefore, the upper bound of the error μ_j of $\hat{r_j}$ is the cumulative value of estimated errors from t_o to present t_j, i.e. $\mu_j = \sum_{\tau=t_o}^{j} \hat{e_\tau}$.

The confidence interval ς is set to ensure the accuracy. Hence, only the error of the estimated $\hat{r_\tau}$ is within the confidence interval, then ς is adoptable, i.e. $\mu_\tau \le \hat{r_\tau} \times \varsigma$ is the adoptable estimated rank.



r
$\hat{r_\iota} + \mu_\iota$
$\hat{r_\iota} + \varsigma \cdot \hat{r_\iota}$
$\hat{r_\iota}$
r_ι
$\hat{r_\iota} - \varsigma \cdot \hat{r_\iota}$
$\hat{r_\iota} - \mu_\iota$
t_o
t_θ
t_ι

FIGURE 4. Setting the duration of radio off period.

The objective of $\max(t_\theta)$ is subject to the accuracy condition. Hence, the value of t_θ is limited to satisfy that $\hat{r}_{t\theta}$ is in the scope of all adoptable estimated ranks. Thus, we obtain $t_\theta = \max(j) - t_o$, where j satisfies $\mu_j \le \hat{r_j} \times \varsigma$. Hence, the rank is also determined as $\hat{r} = \hat{r_j}$.

Figure 4 shows the setting of t_θ and $\hat{r}$ visually by the rank estimation sequence $\hat{r_\tau}$, the cumulative error μ_τ, and the confidence interval ς.

5.4. Storage optimization

We optimize the storage consumption in this subsection. According to CS theory [23–25], if an $n \times t$ matrix has the low-rank feature with rank r, the total $n \times t$ data could be near-optimally recovered by the amount of

$$K \ge C \cdot r \cdot \log_2 \left(\frac{n \cdot t}{r} \right) \tag{12}$$

data, where C is a constant. And the selection of these K data should follow the random distribution in $n \times t$ matrix.

In our design, the period of a cycle includes $t_o + t_\theta$ time slots, where t_o is given and t_θ is obtained by the step of 'Duration Decision'. The rank of the $n \times (t_o + t_\theta)$ matrix is estimated as $\hat{r}$ shown in Fig. 4. To ensure the accuracy, the confidence interval ς is considered to make up the error of estimation. Substitute these variables into Equation (12), we obtain that $K \ge C \cdot \hat{r} \cdot (1+\varsigma) \cdot \log_2(n \cdot (t_o + t_\theta)/\hat{r} \cdot (1+\varsigma))$ data should be randomly gathered for accurate environment reconstruction.

Since the nodes are in active state during t_o period, the amount of gathered data are $n \times t_o$, which is much more than $K \times (t_o/(t_o + t_\theta))$. However, during t_θ period, $M = K \times (t_\theta/(t_o + t_\theta))$ data are required to be gathered. These M data need to be gathered by n nodes in t_θ time slots. Hence, the sensing probability of every node per time slot is $p = M/(n \times t_\theta)$. Substitute M and K into p, we obtain

$$p \ge \frac{1}{n \cdot (t_o + t_\theta)} \cdot C \cdot \hat{r} \cdot (1+\varsigma) \cdot \log_2 \left(\frac{n \cdot (t_o + t_\theta)}{\hat{r} \cdot (1+\varsigma)} \right). \tag{13}$$

In Equation (13), the only unknown parameter is the constant coefficient C. Since the environment usually does not change quickly, we consider that C does not change within one cycle. Then, we can calculate C on the basis of the sample data. We represent Equation (12) by $C = K/r \cdot \log_2(n \cdot t/r)$. Similarly, C_o in the $n \times t_o$ matrix of sample data can be presented by

$$C_o = \frac{K_o}{r_o \cdot \log_2(n \cdot t_o/r_o)}, \tag{14}$$

where r_o is the rank of $n \times t_o$ matrix. In Equation (14), n and t_o are known, r_o can be computed by SVD as Equation (7). Thus, C_o could be obtained after K_o is known.

Then, we introduce the method to search the value of K_o based on the accuracy requirement ϵ_{th}. Assume X_o is denoted the EM of the sample data. B_o is a $(n \times t_o)$ BIM. The number of 0 in B_o is denoted by N_0 and the number of 1 is denoted by $N_1 = n \times t_o - N_0$. Initializing $N_0 = 0$, i.e. B_o is a matrix with all 1's at the beginning. (1) Set $N_0 = N_0 + 1$, the position of the additional 0 is randomly selected in B_o. (2) $G_o = X_o \cdot B_o$ by Definition4. (3) Interpolate G_o by CS algorithm to get $\hat{X}_o$. (4) Compute the ER-Error ϵ_o between X_o and $\hat{X}_o$ by Equation (6). The procedure of (1)–(4) is repeated until $K_o = \min(N_0)$ is found, in which this N_0 matches the condition that $\epsilon_o \leq \epsilon_{\text{th}}$. Substitute this K_o into Equation (14), we obtain the constant coefficient C_o.

Substitute $C = C_o$ into Equation (13), we obtain

$$p \geq \frac{1}{n \cdot (t_o + t_\theta)} \cdot C_o \cdot \hat{r} \cdot (1 + \varsigma) \cdot \log_2\left(\frac{n \cdot (t_o + t_\theta)}{\hat{r} \cdot (1 + \varsigma)}\right). \tag{15}$$

Hence, the value of $\min(p)$ is obtained and this p can guarantee the reconstruction accuracy.

Algorithm 1 REDG algorithm @ sink.

```
Input:
    $t_o$: the duration of radio on in a duty-cycle
    $\varsigma$: the confidence interval of rank estimation
    $\epsilon_{th}$: the threshold of ER-Error
Notation:
    $t_\theta$: the duration of radio off in a duty-cycle
    $p$: the sensing probability of a node per time slot
Main procedure:
 1: gather stored data from $n$ nodes;
 2: $X_o \leftarrow$ gather EC data from $n$ nodes during $t_o$ time slots;
 3: generate $(r_\tau)$ sequence by $X_o$ using SVD, $1 \geq \tau \geq t_o$;
 4: estimate $(\hat{r_\tau})$ sequence by $r_\tau$ using ARIMA, $\tau > t_o$;
 5: estimate $(\mu_\tau)$ sequence by $(r_\tau)$ and $(\hat{r_\tau})$ using ARIMA, $\tau > t_o$;
 6: $t_\theta \leftarrow \max(j) - t_o$, $\hat{r} \leftarrow \hat{r}_{\max(j)}$, subject to $\mu_j \leq \hat{r_j} \times \varsigma$;
 7: search $K_o$ by $X_o$ and $\epsilon_{th}$ using CS;
 8: $C_o \leftarrow K_o/[r_o \cdot \log_2(n \cdot t_o/r_o)]$;
 9: $p \leftarrow \frac{1}{n \cdot (t_o+t_\theta)} \cdot C_o \cdot \hat{r} \cdot (1+\varsigma) \cdot \log_2\left(\frac{n \cdot (t_o+t_\theta)}{\hat{r} \cdot (1+\varsigma)}\right)$;
10: send $t_\theta$ and $p$ to $n$ nodes;
11: wait $t_\theta$ time slots;
```

Algorithm 2 REDG algorithm @ node.

```
Main procedure:
 1: radio on;
 2: if $t_\theta$ and $p$ are received then
 3:     radio off in following $t_\theta$ time slots;
 4:     sense data with probability $p$ and store them until radio on;
 5: else
 6:     send stored data to sink;
 7:     sense and send EC data to sink per time slot;
 8: end if
```

5.5. Algorithm

Three parts of algorithms are needed for carrying out REDG. They are CS algorithm, REDG algorithm at sink side and the REDG algorithm at node side.

First, CS is a mature technology for matrix recovery. Many excellent works have investigated the CS-based algorithms. The CS algorithm adopted in this paper is the same algorithm in [26, 28]. For pseudo-code the reader can refer to Algorithm 1 in [26]. The main function of CS algorithm is to interpolate all missing data in GM and rebuild a estimated matrix E2M to achieve the environment reconstruction in cyberspace.

Secondly, Algorithm 1 presents one cycle of the pseudo-code of REDG algorithm at the sink side. The sink keeps running this Algorithm 1 cycle-by-cycle.

There are three parameters required to set in Algorithm 1: t_o, ς and ϵ_{th}. (1) Since each node senses the environmental data once per time slot, t_o is the amount of sampled data per node during active state. t_o is at least ≥ 3, otherwise, ARIMA methods cannot predict $\hat{r_\tau}$ or μ_τ due to a too short original sequence. We suggest to configure $t_o \geq 24$, which is a usual standard for good performance of ARIMA prediction. (2) We propose to set $\varsigma = 5\%$, as the default setting of confidence interval for prediction in SPSS and SAS. (3) The setting of ϵ_{th} relies on the accuracy requirement of an application. We set ER-Error $\epsilon_{\text{th}} = 5\%$ in our experiment.

The goal of Algorithm 1 is to compute and send the values of t_θ and p to all sensor nodes. The main procedure is to gather the stored data from n nodes when their radios are just on; then to gather t_o sample data from each of n nodes when their radios are still on; to compute t_θ and p on the basis of $n \times t_o$ sample data using SVD, ARIMA and CS; to send t_θ and p to all nodes; finally, to wait t_θ time slots for the next cycle.

Thirdly, Algorithm 2 presents the pseudo-code of REDG algorithm at the node side. The goal of Algorithm 2 is to control the nodes among the states of active, half-active or sleep. The main procedure is to turn on the radio by default, sense and send (i.e. active state) environmental data every time slot. When the values of t_θ and p are received, the node switches and keeps the radio off in the next t_θ time slots. In each of these t_θ time slots, the node senses and stores (i.e. half-active state) environmental data with the probability p. After the t_θ time slots, the node turns on the radio again, sends the stored data to the sink and works as active state again.

The computational complexity of SVD is $O(nt_o)$ [26]; that of CS is $O(\lambda nt_o)$ [28], where λ is a given value of the total number of iterations; and that of ARIMA is $O(n^2t_o^2)$ [29]. Thus, the computational complexity of Algorithm 1 is $O(n^2t_o^2)$. Since Algorithm 1 runs at sink side, the computation capability of current sink device is adequate to compute a $O(n^2t_o^2)$ algorithm quickly. The computational complexity of Algorithm 2 is $O(1)$. In REDG, Algorithms 1 and 2 form a distributed sampling and centralized computing data gathering protocol.

Discussion: If a WSN is required to gather multiple environments (e.g. temperature, light, humidity together) during the same period, REDG algorithm is also adoptable. After gathering the sample data in every cycle, the sink computes t_θ and p of each environment. To ensure the accuracy requirement, the minimum t_θ and the maximum p of these environments are decided to be the parameters for data gathering.

6. PERFORMANCE EVALUATION

We implement a real WSN testbed and conduct trace-driven simulations to evaluate the performance of the proposed REDG protocol.

6.1. Experimental implementation

Experimental testbed: Our testbed includes totally 51 TelosB sensor nodes. They are divided into three groups: Group A, B and C, carrying out different data gathering protocols for comparison. Each group has 16 nodes to sense environmental data and 1 sink node to gather these data. In a 7.2 m × 6 m open-air area, $4 \times 4 = 16$ positions are selected to deploy sensor nodes as a grid. The transversal distance between two neighbor positions is 2.4 m and the longitudinal distance between two neighbor positions is 2 m. As shown in Fig. 5, at each position, there are three sensor nodes, which belong to three groups, respectively. Totally 48 nodes are deployed in the area. The other three sink nodes connect to three laptops as shown in Fig. 5.

Each group of 17 sensor nodes organizes its own network. These nodes transmit data using IEEE 802.15.4 Zigbee standard. There are 16 ZigBee channels in the 2.4 GHz ISM band, with each channel requiring 5 MHz of bandwidth. The transmission rate is up to 250 kbps. The three groups of WSNs work during the same period with three non-overlap channels, so there are no interference among them.

Implementation setting: There are some common configurations for three groups. The duration of every time slot is set as 1 min. The node senses data once per minute when at the active time slot. The gathered data are stored in database in the laptop according to our custom format, including sensing time stamp, node ID, temperature, humidity, light, voltage and RSSI. Each record costs 160 Bytes storage space.

Then, we set the three groups of WSNs, respectively. Group A: CTP. TinyOS library provides the code of CTP. The radio is always on for sending, receiving or forwarding data. Group B: Fixed low-duty-cycle (FLDC12.5), one cycle is set 4 h with 30-min active state and 3.5-h sleep state. So the duty cycle is 12.5%. (We also tested the cases that the duty cycles of FLDC are set to 50, 25 and 6.25%. Compared with FLDCs of other duty cycles, the performance of FLDC12.5 has the least duty-cycle subject to ER-Error $\leq$ 5%.) Group C: REDG. The parameters are set as $t_o = 30$, $\varsigma = 5\%$ and $\epsilon_{\text{th}} = 5\%$.

Experiment results: To measure the performance, we compare the reconstruction accuracy, energy consumption and storage consumption among three protocols. These three protocols run 10 days for temperature reconstruction, 10 days for humidity reconstruction and 10 days for light reconstruction.

The metric of accuracy is the ER-Error, which is defined in Metric 1 and can be computed by Equation (6). Figure 6a plots the histogram of ϵ in different group in different environments. Group A: Since CTP gathers all environmental data, there is no ER-Error $\epsilon = 0$. Group B: FLDC12.5 loses 87.5% environmental data. Although the part of missing data are estimated by CS algorithm to rebuild the dynamic environment, ER-Errors are 3.1% in temperature, 3.6% in humidity and 5.4% in light. Group C: REDG offers the satisfied results on ER-Errors. Since the prediction of t_θ is controlled by $\varsigma = 5\%$, and the amount of gathered data is controlled by $\epsilon_{\text{th}} = 5\%$, REDG displays $\epsilon = 1.8\%$ in temperature, 2.1% in humidity and 4.7% in light after CS. In summary, the comparison result indicates that the REDG algorithm can ensure the accuracy requirement.

The energy consumption is measured by the average duty cycle α of sensor nodes, i.e. the ratio of the total duration of radio on period in 10 days. The results of average duty cycle are displayed in Fig. 6b. Group A: The radio keeps turning on in CTP, so $\alpha = 100\%$. Group B: Since the duty cycle is fixed in

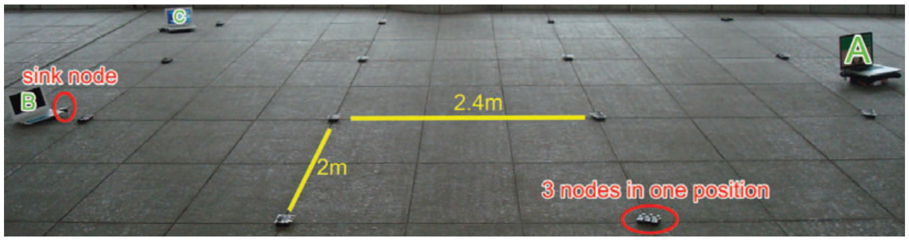


FIGURE 5. Experimental testbed.

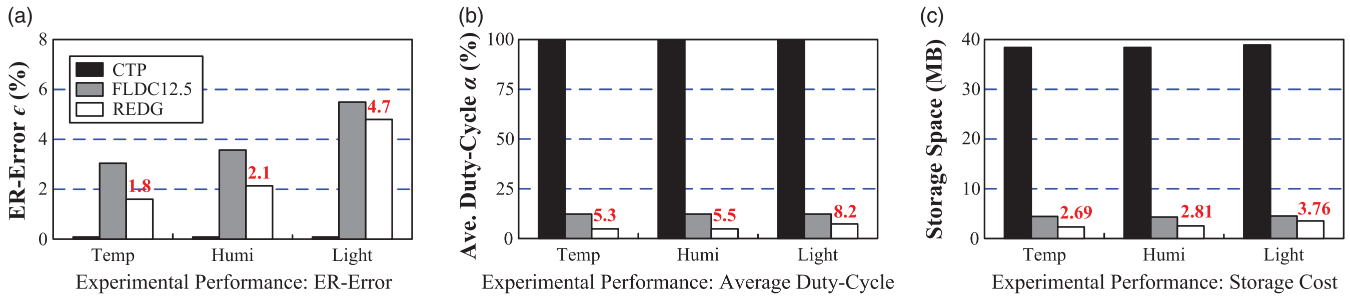


FIGURE 6. Performance evaluation.

FLDC, $\alpha = 12.5\%$. Group C: the duty cycle in REDG changes according to the dynamic environment. From Fig. 6b, we find that $\alpha = 5.3\%$ in temperature, $\alpha = 5.5\%$ in humidity and $\alpha = 8.2\%$ in light. The results imply REDG is better than FLDC12.5 and much better than CTP on energy saving in our experiment.

The amount of gathered data in database in the laptops is used to measure the storage consumption. Figure 6c illustrates the storage costs among different protocols. Group A: We can easily derive the theoretical storage consumption for CTP: the period of data gathering is 10 days; there are 16 nodes reporting the environmental data to the sink; the data are sent 1 per minute per node; each record costs 160B storage space; there are totally $10 \times 24 \times 60 \times 16 \times 1 = 230\,400$ records, ∼36.86 MB. In the real experiment, the storage consumption of CTP is near 37.22 MB, which is close to the theoretical result. Group B: FLDC12.5 senses data only in the active state, so the amount of gathered data is also nearly 12.5%. The theoretical space costs are 4.61 MB. From Fig. 6c, the space costs are ∼4.83 MB in our experiment. Group C: In REDG, the parameter p is adapted with the dynamic environment. Hence, we cannot derive a theoretical result of storage consumption for REDG. And the experimental results are displayed in Fig. 6c, which are 2.69 MB for temperature, 2.81 MB for humidity, 3.76 MB for light. In conclusion, REDG saves much more storage space than CTP and FLDC12.5.

REDG outperforms the current data gathering methods in this experiment. Compared with CTP, REDG is much better on resource efficiency within the accuracy requirement. Compared with FLDC12.5, both of them can achieve the accuracy requirement, but REDG performs much better on energy and storage consumption.

6.2. Trace-driven simulation

Simulation overview: Although the experiment verifies the practice and efficiency of REDG, it only carries out in an experimental scenario with some limitations such as the small-scale WSN (17 nodes for every group), small area (7.2 m × 6 m), one scenario (open-air space) and one certain sensing frequency (once per minute). To test the extensive applicability of REDG, we conduct the simulations based on the diverse datasets from real applications. Table 1 lists the three datasets. These three datasets are in diverse scales (50, 249 and 15 nodes), diverse areas (40 m × 30 m, 200 m × 100 m and 300 m × 100 m), diverse scenarios (indoor, forest and ocean) and diverse sensing interval (0.5, 5 and 2 min).

These datasets are also divided into 6 EMs: Indoor-temp, indoor-light, forest-temp and etc. Every EM is simulated by CTP, FLDC50, FLDC25, FLDC12.5, FLDC6.25 and REDG. The metric of accuracy is still measured by ER-Error ϵ, and the energy consumption is still measured by average duty cycle α. However, the storage formats for diverse datasets are different. To use a unified metric, we adopt the number of records to measure the storage consumption in our simulation.

Simulation setting: Since the radios are assumed to keep turning on in CTP, all data are gathered. Hence, GM G is actually the same as EM X. There is no ER-Error. The average duty cycle is 100%. The number of records is equal to $n \times t$.

For FLDC, one cycle is set 240 slots. We use FLDC12.5 as an example to explain the procedure of FLDC simulation. The active state of a node is set to 30 slots and the subsequent 210 slots are at the sleep state for this node. Consequently, BIM is a $n \times t$ matrix with every 240 columns as cycle. In one cycle, the values in first 30 columns are all 1, and the values in the rest 210 columns are all 0:

$$B = \begin{bmatrix} 1 & \cdots & 1 & 0 & \cdots & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \cdots & 1 & 0 & \cdots & 0 & 1 & \cdots \end{bmatrix}_{n\times t} . \quad (16)$$
$$\leftarrow 30 \rightarrow \quad \leftarrow 210 \rightarrow$$

Then, all the gathered data can be presented by GM $G = X \cdot B$. And E2M $\hat{X}$ can be estimated by CS algorithm based on GM G. Compared $\hat{X}$ and X, the ER-Error can be computed by Equation (6). The average duty cycle is $\frac{30}{240} = 12.5\%$. The number of records is equal to $(n \times t \times 12.5\%)$.

For REDG, the simulations are conducted according to the Algorithms 1 and 2 with the setting of $t_o = 30$, $\varsigma = 5\%$ and $\epsilon_{\text{th}} = 5\%$. So that a GM G can be obtained from the datasets. Then, we can estimate $\hat{X}$ by CS and compute ϵ by Equation (6). The average duty cycle can be computed by the statistics of the

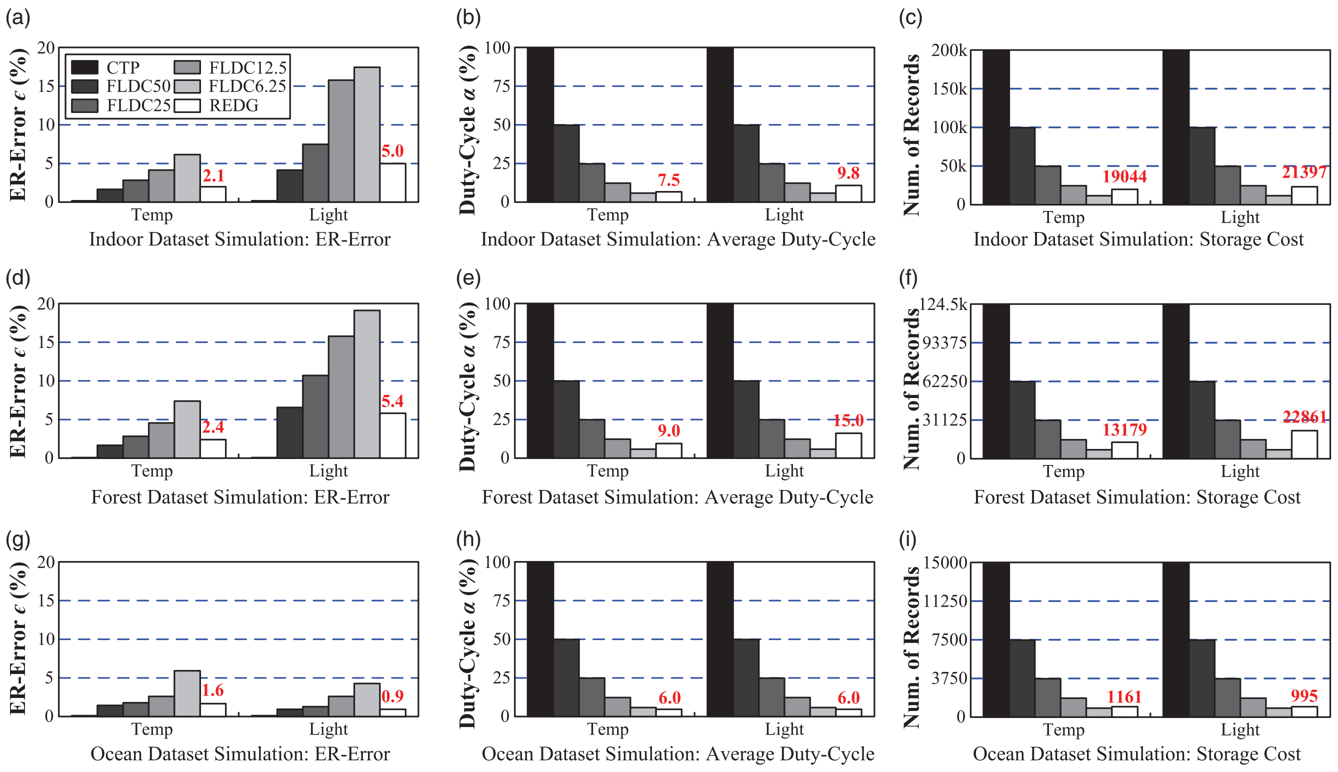


FIGURE 7. Performance simulation.

total number of slots for active slots. The number of records is equal to the number of non-zero elements in G.

Simulation results: Figure 7 shows the ER-Errors, the average duty cycle, and the number of records of different data gathering protocols among indoor, forest and ocean datasets in our simulations.

We can find in Fig. 7a, d, g, every ϵ is 0 for in CTP; the smaller of the duty cycles, the larger ER-Errors in FLDC are; and the ER-Errors of REDG are all <5% no matter which scenario or environment.

The average duty cycles of CTP, FLDC50, FLDC25, FLDC12.5, FLDC6.25 are 100, 50, 25, 12.5, and 6.25 in Fig. 7b, e, h. These values are fixed, which are independent of scenarios or environments. Nevertheless, the average duty cycles of REDG are dynamic corresponding to the diverse scenarios or environments. Most average duty cycles α of REDG are smaller than 10%. For example, α in Ocean-Temp and Ocean-Light are only 6.0% and in Indoor-Temp is 7.5%. The worst case in the simulations is the Forest-Light that $\alpha = 15\%$.

In Fig. 7c, f, i, the numbers of records gathered by CTP and FLDC are also fixed, which is equal to $n \times t \times \alpha$. The numbers of records gathered by REDG in Indoor-Temp, Indoor-Light, Forest-Temp, Forest-Light, Ocean-Temp, Ocean-Light are 19 044, 21 397, 13 179, 22 861, 1161, 995, respectively, which are usually smaller than 12% of the total number of data records.

In summary, the results in the simulation are similar to the performance in the experiment. The proposed REDG protocol guarantees the reconstruction accuracy with low energy and storage consumption. Compared with CTP, REDG saves much more energy and storage resources with only a little accuracy loss. Compared with FLDC, if considering with the accuracy requirement, REDG performs better on energy and storage efficiency; if considering with the nearly same resource cost, REDG can rebuild more accurate environment than FLDC. The most important property of REDG is that its parameters (i.e. the duty cycle and the amount of gathered data) are self-adaptive to the dynamic environment.

7. CONCLUSION

We proposed a novel data gathering protocol in WSNs for environment reconstruction, which jointly considers the resource efficiency and the reconstruction accuracy issues. This protocol can adapt the duty cycle and the sensing probability of sensor nodes according to the dynamic environment based on the CS theory and time series prediction. We implemented this protocol in a real WSN testbed and conducted extensive trace-driven simulations. The evaluation results demonstrate that the proposed protocol dramatically reduces energy and storage consumption compared with conventional data gathering protocols while the accuracy of environment reconstruction is guaranteed.

FUNDING

This research was supported by Chinese 973 Program (2013CB035503), China NSF grants No. 61170296, No. 61190125, No. 61303202, No. 61073158, No. 61100210, No. 61300174, Chinese R&D Program (2013BAH35F01), CPSF 2013M530511, US NSF grants CNS-1117438, Singapore-MIT IDC IDD61000102a, SUTD-ZJU/RES/03/2011, NRF2012EWT-EIRP002-045, STCSM Project No. 12dz1507400, No. 13511507800.

REFERENCES

[1] Kong, L., Jiang, D. and Wu, M.-Y. (2010) Optimizing the Spatio-Temporal Distribution of Cyber-Physical Systems for Environment Abstraction. *Proc. IEEE ICDCS 10*, Genoa, Italy, June 21–25, pp. 179–188. IEEE Press, Piscataway, NJ, USA.

[2] He, T., Krishnamurthy, S., Stankovic, J.A., Abdelzaher, T.F., Luo, L., Stoleru, R., Yan, T., Gu, L., Hui, J. and Krogh, B.H. (2004) Energy-Efficient Surveillance System Using Wireless Sensor Networks. *Proc. ACM MOBISYS 04*, Boston, MA, USA, June 6–9, pp. 270–283. ACM Press, New York, NY, USA.

[3] Vasilescu, I., Kotay, K., Rus, D., Dunbabin, M. and Corke, P. (2005) Data Collection, Storage, and Retrieval with an Underwater Sensor Network. *Proc. ACM SENSYS 05*, San Diego, CA, USA, November 2–4, pp. 154–165. ACM Press, New York, NY, USA.

[4] Mo, L., He, Y., Liu, Y., Zhao, J., Tang, S., Li, X. and Dai, G. (2009) Canopy Closure Estimates with GreenOrbs: Sustainable Sensing in the Forest. *Proc. ACM SENSYS 09*, Berkeley, CA, USA, November 4–6, pp. 99–112. ACM Press, New York, NY, USA.

[5] Liu, G., Tan, R., Zhou, R., Xing, G., Song, W.-Z. and Lees, J. (2013) Volcanic Earthquake Timing using Wireless Sensor Networks. *Proc. ACM IPSN 13*, Philadelphia, PA, USA, April 8–11, pp. 91–102. ACM Press, New York, NY, USA.

[6] Werner-Allen, G., Lorincz, K., Johnson, J., Lees, J. and Welsh, M. (2006) Fidelity and Yield in a Volcano Monitoring Sensor Network. *Proc. USENIX OSDI*, Seattle, WA, USA, November 6–8, pp. 381–396. USENIX Association, Berkeley, CA, USA.

[7] Ji, S. and Cai, Z. (2013) Distributed data collection in large-scale asynchronous wireless sensor networks under the generalized physical interference model. *IEEE/ACM Trans. Netw.*, **21**, 1270–1283.

[8] Liu, Y., Zhang, Q. and Ni, L. (2010) Opportunity-based topology control in wireless sensor networks. *IEEE Trans. Parallel Distrib. Syst.*, **21**, 405–416.

[9] He, L., Cheng, P., Gu, Y., Pan, J., Zhu, T. and Liu, C. (2014) Mobile-to-Mobile Energy Replenishment in Mission-Critical Robotic Sensor Networks. *Proc. IEEE INFOCOM 14*, Toronto, Canada, April 27–May 2. IEEE Press, Piscataway, NJ, USA.

[10] Baraniuk, R. (2011) More is less: signal processing and the data deluge. *Science*, **331**, 717–719.

[11] Dam, T. and Langendoen, K. (2003) An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. *Proc. ACM SENSYS 03*, Los Angeles, CA, USA, November 5–7, pp. 171–180. ACM Press, New York, NY, USA.

[12] Uysal-Biyikoglu, E., Prabhakar, B. and Gamal, A.E. (2002) Energy-efficient packet transmission over a wireless link. *IEEE/ACM Trans. Netw.*, **10**, 487–499.

[13] Xu, Y., Heidemann, J. and Estrin, D. (2001) Geography-Informed Energy Conservation for Ad Hoc Routing. *Proc. ACM MOBICOM 01*, Rome, Italy, July 16–21, pp. 70–84. ACM Press, New York, NY, USA.

[14] Ye, W., Heidemann, J. and Estrin, D. (2002) An Energy-Efficient MAC Protocol for Wireless Sensor Networks. *Proc. IEEE INFOCOM 02*, New York, NY, USA, June 23–27, pp. 1567–1576. IEEE Press, Piscataway, NJ, USA.

[15] Gu, Y. and He, T. (2007) Data Forwarding in Extremely Low-Duty-Cycle Sensor Networks with Unreliable Communication Links. *Proc. ACM SENSYS 07*, Sydney, Australia, November 6–9, pp. 321–334. ACM Press, New York, NY, USA.

[16] Guo, S., Gu, Y., Jiang, B. and He, T. (2009) Opportunistic Flooding in Low-Duty-Cycle Wireless Sensor Networks with Unreliable Links. *Proc. ACM MOBICOM 09*, Beijing, China, September 20–25, pp. 133–144. ACM Press, New York, NY, USA.

[17] Xiong, S., Li, J., Li, M., Wang, J. and Liu, Y. (2011) Multiple Task Scheduling for Low-Duty-Cycled Wireless Sensor Networks. *Proc. IEEE INFOCOM 11*, Shanghai, China, April 10–15, pp. 1323–1331. IEEE Press, Piscataway, NJ, USA.

[18] Luo, C., Wu, F., Sun, J. and Chen, C.W. (2009) Compressive Data Gathering for Large-Scale Wireless Sensor Networks. *Proc. ACM MOBICOM*, Beijing, China, September 20–25, pp. 145–156. ACM Press, New York, NY, USA.

[19] Gupta, H., Navda, V., Das, S. and Chowdhary, V. (2005) Efficient gathering of correlated data in sensor networks. *ACM Trans. Sensor Netw.*, **4**, article no. 4.

[20] Yu, Y., Krishnamachari, B. and Prasanna, V.K. (2004) Energy-Latency Tradeoffs for Data Gathering in Wireless Sensor Networks. *Proc. IEEE INFOCOM 04*, Hong Kong, China, March 7–11. IEEE Press, Piscataway, NJ, USA.

[21] Intel Lab Data: http://www.select.cs.cmu.edu/data/labapp3/index.html.

[22] Yang, Z., Li, M. and Liu, Y. (2007) Sea Depth Measurement with Restricted Floating Sensors. *Proc. IEEE RTSS 07*, Tucson, AR, USA, December 3–6, pp. 469–478. IEEE Press, Piscataway, NJ, USA.

[23] Candes, E., Romberg, J. and Tao, T. (2006) Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inf. Theory*, **52**, 489–509.

[24] Candes, E. and Tao, T. (2006) Near optimal signal recovery from random projections: universal encoding strategies? *IEEE Trans. Inf. Theory*, **52**, 5406–5425.

[25] Donoho, D. (2006) Compressed sensing. *IEEE Trans. Inf. Theory*, **52**, 1289–1306.

[26] Li, Z., Zhu, Y., Zhu, H. and Li, M. (2011) Compressive Sensing Approach to Urban Traffic Sensing. *Proc. IEEE ICDCS 11*, Minneapolis, MN, USA, June 20–24, pp. 889–898. IEEE Press, Piscataway, NJ, USA.

[27] Rallapalli, S., Qiu, L., Zhang, Y. and Chen, Y. (2010) Exploiting Temporal Stability and Low-Rank Structure for Localization in Mobile Networks. *Proc. ACM MOBICOM 10*, Chicago, IL, USA, September 20–24, pp. 161–172. ACM Press, New York, NY, USA.

[28] Zhang, Y., Roughan, M., Willinger, W. and Qiu, L. (2009) Spatio-Temporal Compressive Sensing and Internet Traffic Matrices. *Proc. ACM SIGCOMM 09*, Barcelona, Spain, August 17–21, pp. 267-278. ACM Press, New York, NY, USA.

[29] Box, G. and Jenkins, G. (1994) *Time Series Analysis: Forecasting and Control*. Prentice-Hall, Englewood Cliffs, NJ.

[30] Kong, L., Xia, M., Liu, X.-Y., Wu, M.-Y. and Liu, X. (2013) Data Loss and Reconstruction in Sensor Networks. *Proc. IEEE INFOCOM 13*, Turin, Italy, April 14–19, pp. 1654–1662. IEEE Press, Piscataway, NJ, USA.

[31] Gnawali, O., Fonseca, R., Jamieson, K., Moss, D. and Levis, P. (2009) Collection Tree Protocol. *Proc. ACM SENSYS 09*, Berkeley, CA, USA, November 4–6, pp. 1–14. ACM Press, New York, NY, USA.

[32] Wang, X., Wang, X., Liu, L. and Xing, G. (2013) DutyCon: a dynamic duty-cycle control approach to end-to-end delay guarantees in wireless sensor networks. *ACM Trans. Sensor Netw.*, **9**, article no. 42.

[33] Bajwa, W., Haupt, J., Sayeed, A. and Nowak, R. (2006) Compressive Wireless Sensing. *Proc. ACM IPSN 06*, Nashville, TN, USA, April 19–21, pp. 134–142. ACM Press, New York, NY, USA.

[34] Chou, C. Ignjatovic, A. and Hu, W. (201) Efficient computation of robust average of compressive sensing data in wireless sensor networks in the presence of sensor faults. *IEEE Trans. Parallel Distrib. Syst.*, **24**, 1525–1534.

[35] Li, P., Hastie, T.J. and Church, K.W. (2006) Very Sparse Random Projections. *Proc. ACM KDD 06*, Philadelphia, PA, USA, August 20–23, pp. 287–296. ACM Press, New York, NY, USA.

[36] Wang, W., Garofalakis, M. and Ramchandran, K. (2007) Distributed Sparse Random Projections for Refinable Approximation. *Proc. ACM IPSN 07*, Cambridge, MA, USA, April 25–27, pp. 331–339. ACM Press, New York, NY, USA.

[37] Lakhina, A., Papagiannaki, K., Crovella, M., Diot, C., Kolaczyk, E. and Taft, N. (2004) Structural Analysis of Network Traffic Flows. *Proc. ACM SIGMETRICS 04*, New York, NY, USA, June 12–16, pp. 61–72. ACM Press, New York, NY, USA.

[38] Anderson, O.D. (1976) *Time Series Analysis and Forecasting: the Box–Jenkins Approach*. Butterworths, London edition.

[39] Papagiannaki, K., Taft, N., Zhang, Z.-L. and Diot C. (2003) Long-Term Forecasting of Internet Backbone Traffic: Observations and Initial Models. *Proc. IEEE INFOCOM 03*, San Francisco, CA, USA, March 30–April 3, pp. 1178–1188. IEEE Press, Piscataway, NJ, USA.