Energy-efficient On-demand Active Contour Service for Sensor Networks

Yangfan Zhou* Junjie Xiong* Michael R. Lyu* Jiangchuan Liu† Kam-Wing Ng*
*Dept. of Computer Science & Engineering, The Chinese Univ. of Hong Kong, Shatin, Hong Kong
†School of Computing Science, Simon Fraser Univ., Burnaby, BC, Canada

Abstract

Contour mapping is an important technique for Wireless Sensor Networks (WSNs) in environmental monitoring to abstract the information of a monitored field. State-of-the-art approaches for contour mapping, however, are neither energy-optimized, nor capable of handling heterogeneous user requests. In this paper, we develop a novel energy-efficient On-demand Active Contour Service (OACS) for power-constrained WSNs. OACS regresses the field intensity function with kernel Support Vector Regression (SVR), a novel machine learning tool that flexibly handles both contour line and contour map requests. OACS also adaptively accommodates a wide range of contour line/map precision requirements: (1) For applications of low precision, only a minimum set of nodes are scheduled in working mode while others are sleeping for conserving energy. (2) For applications of high precision, through an active and progressive learning algorithm, OACS determines the best set of nodes that should be turned on for improving the contour line/map precision. Evaluation based on diverse realistic models demonstrates that OACS provides quality and seamless contour services for various application requirements yet significantly conserves energy.

1. Introduction

In-situ sensing with wireless sensor networks (WSNs) is a promising technique for environmental monitoring applications [1, 22]. In these applications, the major task of a WSN is to measure, process, and convey the sensor data to a data sink so that the sink can reconstruct the information of a scalar field of interest, *e.g.*, the temperature distribution throughout a monitored space, or the boundary where the concentration of a toxic gas reaches a dangerous level. Since such a field is continuous in nature, *contour mapping* turns out to be a natural and necessary service: Based on the in-situ sensor readings, the sinks of WSNs can construct contour lines or contour maps to present the information of the scalar fields. A *contour line* is a line along which the intensity of the field is a constant value. The value is hence called *the value of the contour line*. A *contour map* is a map illustrated with a set of such contour lines.

Contour mapping has long been recognized as an important approach for WSN applications [18], *e.g.*, to locate and monitor an event of interest [31] and to capture the WSN system life conditions such as the energy levels of the nodes [33]. It can also provide instant and user-friendly visualization of the scalar field of interest [17, 30].

One challenge faced by contour mapping in WSNs is how to handle diverse contour requests from users. A user may request a single contour line to obtain the information of a boundary, or request a contour map to obtain the information of the entire field of interest. Moreover, the user may also have different requirements on the precision of a contour line/map. For example, he/she may be interested in observing more details of the field when a particular event takes place and hence may require the network to produce a finer map. However, a contour service that can handle such diverse user requests with energy-efficiency is still at large in the current state of the art.

The existing contour mapping approaches are generally optimized for either contour line (*e.g.*, [17, 25, 30]) or contour map (*e.g.*, [18, 23]) requests, but not for both. More importantly, they largely ignore the fact that the users may have different requirements on the precision of a contour line/map. Their focus is to minimize the number of packets that should be reported to the sink for achieving energy efficiency. They generally require each in-network node to sense the environmental data of interest, although the sensor readings are processed in the network and some are possibly suppressed. As a result, they are essentially *best-effort* approaches and lack a scheme for tuning the precision of the line/map by controlling the number of working nodes [9, 17, 18, 25, 30]. Such precision tuning, however, is not just a marginal service consideration, but very crucial for energy saving. With such a scheme, we can put some in-network nodes into sleeping mode and thus save more energy when low precision is acceptable.

To address this critical challenge, we develop a novel energy-efficient contour service, namely, On-demand Ac-

978-1-4244-5113-5/09/$25.00 ©2009 IEEE

tive Contour Service (OACS), for WSNs to seamlessly handle diverse user requests. OACS first divides the network area into clusters. It then regresses a *field intensity function* in each cluster by taking sensor readings with sensor locations as input samples. This provides the flexibility to handle contour line and contour map requests adaptively. For a contour line request, what we focus is to let the regression result accurately represent the reality in the area around the contour line. Hence, only the sensor readings close to the contour line value are required for the regression. Whereas for contour map requests, since the contour lines of interest are spreading throughout the network area, the regression result is important at everywhere of the network. Hence more sensor readings can be involved in the regression.

To deal with different requirements on contour line/map precision, OACS initially requires only a minimum set of nodes on duty while putting the rest in sleeping mode. According to the precision requirement, OACS then progressively activates sleeping nodes to enhance the precision expected by the users. To this end, it incorporates a comprehensive set of selection algorithms for the nodes-to-be-activated, which are enlightened by the recent advances in *active learning* [28]. This novel two-step design of OACS seamlessly accommodates request diversity yet significantly conserves energy.

The rest of this paper is organized as follows. Section 2 studies the related work. In Section 3, we provide a system-level overview of our contour service. Section 4 analyzes the contour mapping problem and a two-step regression approach is illustrated. Section 5 reports the performance study results and Section 6 provides the conclusion remarks.

2. Related Work

Contour mapping has long been studied by geography and geology researchers [6, 8], where their concern is mainly on how to generate a contour map with all data at hand. A straightforward application to WSNs is that the sink collects all individual sensor readings and then builds the map centrally (*e.g.*, with linear interpolation such as Kriging [15]). Such an approach, although simple, incurs too much energy consumption as every node has to sense and report data to the sink. It does not fit WSNs due to their energy constraints. This leads to the design of many *distributed* approaches. Some propose that only the nodes with readings close to the value of the contour lines report their readings to the sink (*e.g.*, [17, 25]). In other approaches, a network is divided into clusters. In each cluster, segments of contour lines are constructed at its cluster head and reported to the sink instead of all the sensor readings(*e.g.* [30]).

The idea of using machine learning techniques in contour mapping is recently suggested in [30]. It formulates the problem of constructing a segment of a contour line as a non-linear classification problem. A severe algorithmic inadequacy of this approach is that it relies heavily on the shape of the distribution of the node's location, while disregarding the sensor readings. As a result, it may generate biased contour lines and generate the same curve for the contour lines with different values. In addition, The application of linear regression to contour mapping is proposed in [17]. But a contour line is non-linear in nature. Such a linear approximation generates results that are either inaccurate or with too many parameters. Moreover, a mobility-assisted approach is proposed in [26], which studies how to design the tracks of a set of mobile nodes to detect a contour line. The problem context, however, is quite different from that studied in this paper.

Note that the above schemes are essentially optimized for calculating a single contour line. When an entire contour map is required, they usually need to construct each contour line separately and eventually the resulting lines are combined into one map [17, 25, 30]. This is actually very inefficient. Neighboring contour lines usually exhibit space correlations, and hence it is possible to construct and represent them in a better way. Taking this into account, many approaches try to exploit such spatial correlations in building an entire map. The idea to abstract a field with isobars (*i.e.*, rectangle area where the field intensity is close to a value) is suggested in [20] for boundary detection, and Hellerstein *et al* propose to build such an isobar map based on a routing tree [12]. Gandhi *et al* suggest summarizing a contour map with a topologically equivalent family of polygons [9]. Silberstein *et al* [23] and Meng *et al* [18] investigate algorithms for spatial and temporal suppression . Their focus is a data aggregation scheme for minimizing energy consumption of communication. Similarly for efficient data management, Guestrin *et al* propose to compress spatiotemporally correlated sensor data with a distributed regression [11]. These approaches, however, are not efficient for calculating a single contour line since they aim at abstracting *all* in-network sensor readings.

To the best of our knowledge, no comprehensive yet adaptive treatment to both contour line and contour map requests is presented in the literature. Moreover, the existing approaches are best-effort only: There is no mechanism for energy conserving when lower precision contour line/map is acceptable. Finally, they lack a proper scheme to actively select sensor nodes in calculating contour lines/maps.

3. System-level Overview of On-demand Active Contour Service

This section provides an overview of the OACS approach in a system perspective. We first identify its working environments. Then we discuss the user requirements and how we encode them in OACS's user enquiries. Finally, we brief

OACS's major work flow in processing its user enquiries.

3.1. Preliminaries

We consider a WSN consisted of N stationary sensor nodes $\{s_j\}_{j=1}^{N}$ randomly deployed in a 2-dimensional network area ϕ (*i.e.*, $\phi \subset \mathbb{R}^2$). Like work in [30], we consider the network consists of K clusters and there is a head h_i for each cluster i [4, 32]. Cluster heads are in charge of constructing the segments of contour lines/maps inside their clusters and reporting the results to the sink. They are generally equipped with a high computational capability hardware platform, *e.g.*, the Crossbow imote2 [5]. Cluster members, on the other hand, can be relatively low-capability nodes such as micaz motes. Let $\mathcal{S}_i$ denote the set of the nodes in cluster i, *i.e.*, s_j is in cluster i if and only if $j \in \mathcal{S}_i$. n_i is the node number in $\mathcal{S}_i$. L_j denotes the location of s_j. Location-awareness is a basic requirement for contour mapping approaches [9, 17, 18, 25, 30, 31]: We consider that each node is aware of its own location, which can be obtained by a light-weighted localization approach (*e.g.*, Calamari [29] distributed with TinyOS [27]). Cluster heads thus know the locations of their cluster members.

Suppose among all nodes in a cluster, there are a portion of *on-duty nodes*, *i.e.*, those that are not in sleeping mode, which maintain the normal sensing task of the network[1]. Let λ denote the percentage of the on-duty nodes in a cluster. We assume that there is a mechanism for cluster heads to let any of its sleeping cluster members change to working mode. This can be done via an active scheme like that proposed in [10] with which a node can be turned on by radio, or a passive scheme in which a sleeping node periodically turns itself on and enquires whether it should change to working mode.

z_j denotes the sensor reading of node s_j when s_j is working. We consider that $z_j \in \mathbb{R}$, *i.e.*, z_j is a 1-dimensional scalar value, *i.e.*, the measured intensity of *one* scalar field. This is assumed without loss of generality: If there are multiple co-existing fields, z_j is then multi-dimensional. Then the contour maps are multiple, each corresponding to one dimension.

3.2. Contour Enquiries

We consider two natural user requirements of contour mapping service. The first one is that a user wants to obtain one single contour line with a given value. Identifying an event boundary (*e.g.*, the boundary of a pollution area as considered in [26]) is an example. The second one is that the user requires a contour map of the whole network area, for example, for visualization of the information of the entire field of interest. Hence, user enquiries are divided into two types, namely, line-enquiries and map-enquiries (*L-enquiries* and *M-enquiries*). In an L-enquiry, a user should give the value of the requested contour line. While in an M-enquiries, since the user requests the network to abstract the *entire* field information, no specific values are required.

Another important issue is the precision of the contour line/map a user requests. During a WSN system run-time, a user may require contour lines/maps with different precisions. For example, a user may ask the system to generate some coarse-grained contour maps periodically for logging the changes of a field. Whereas during a particular time interval (*e.g.*, when an event takes place), the user may need a finer map for more details of the field.

Since neither the *actual* field intensity nor even its distribution is *a priori* knowledge, it is impossible to adopt traditional precision metrics such as mean square error (MSE) for presenting the precision of a calculated contour line/map: For example, if a user needs a contour line with its value being 10 and MSE being 0.5. After calculating a contour line with sensor readings, no one can tell whether its MSE satisfied the requirement or not since there is no way to know the ground-truth contour line. This poses a challenging problem: The precision requirement of an enquiry has to be defined in a way that it should not only make sense of how well a contour line/map matches the reality but also be a calculable value.

Note that the most precise contour line/map segment a cluster head h_i can get is the one constructed based on the readings of all nodes in $\mathcal{S}_i$. A contour service can generally provide a more precise result if it takes the readings of more nodes into account in constructing a contour line/map. Hence, the number of the involved nodes is naturally a good index to the precision of the result. The more the number of nodes involved is, the more precise the result is, whereas the more energy is required since more nodes are needed to sense and report their readings. Tuning the number of involved nodes provides the flexibility for a user to trade off energy and the precision of the result.

Hence, the formats of enquiries are as follows. An L-enquiry is a 2-tuple L=$[cv, p]$ and an M-enquiry is denoted by M=$[p]$, where cv is the value of the requested contour line, and p is the precision requirement, which is the percentage of the sensor node number in each cluster that the user requires to build a contour line/map.

3.3. Service Overview

Our contour service is an on-demand service: A user-enquiry can be directly broadcast to the sensor nodes by the sink with its powerful antenna, or can be broadcast through multi-hop. When receiving an M-enquiry, all work-

[1]The roles of being on-duty and being sleep can be rotated for all in-network nodes based on a node grouping mechanism (*e.g.*, [16, 34]) to balance the residual energy of the nodes or to maintain coverage.

ing nodes report their readings to their cluster heads, while when receiving an L-enquiry $[cv, p]$, only those working nodes whose readings are close to the contour value cv report their readings to their cluster heads.

Each cluster head then knows the intensity of the field at a set of in-network locations. With these data, it computes the contour line/map segment inside its corresponding cluster with a Support Vector Regression (SVR) kernel-based regression technique.

It then examines whether the precision requirement of the results matches the user requirements p with the initial working nodes. If the answer is *no*, it will employ an active machine-learning based algorithm to select a set of sleeping nodes, turn them on, obtain their readings, and refine the results of the kernel SVR scheme, until the results are satisfactory. Finally, after the results are reported to the sink for combining into an overall contour line/map, the additional working nodes go back to sleep mode.

4. Regression-based Contour Mapping Algorithms

Since the cluster heads play important roles in our contour service, in this section we focus on how a cluster head constructs the part of the contour line/map that is located in its cluster.

4.1. Contour Mapping with Kernel SVR

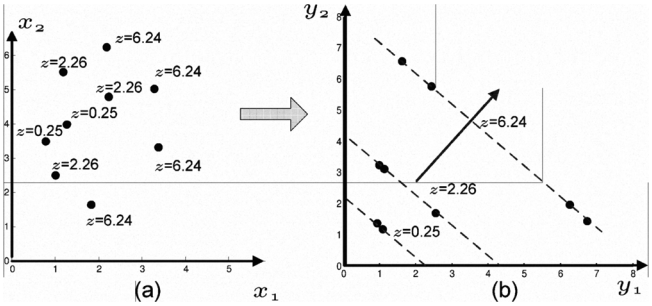
The intensity z of a scalar field is essentially an unknown continuous function $f(\mathbf{x})$ defined on ϕ, where $\mathbf{x}$ is a 2-dimensional vector denoting a location in the network area:

$$z = f(\mathbf{x}) \ \ (\mathbf{x} \in \phi). \tag{1}$$

An actual contour line with value cv is then a curve where the points on the curve satisfy $f(\mathbf{x}) = cv$. While an entire contour map is composed by a lot of such curves (with values forming an arithmetic sequence) spreading throughout the network area. Given an estimation of the function $f(\mathbf{x})$, we can easily find out a contour line/map.

The sensor readings can be deemed as the sample values of the function $f(\mathbf{x})$ where the nodes locate. Note that these sensor readings, however, are subject to measurement errors due to the influence of noise. Hence, contour mapping is essentially a procedure we regress $f(\mathbf{x})$ based on these noisy sample values.

Formally, for each cluster i, give h_i a set of samples, *i.e.*, the mappings $\{L_k \mapsto z_k\}_{\forall k \in \mathcal{W}_i}$ where $\mathcal{W}_i$ is a subset of all nodes in $\mathcal{S}_i$. Its goal is a function $\widetilde{f}_i(\mathbf{x})$ to estimate $f(\mathbf{x})$ in its cluster area, which satisfies $\widetilde{f}_i(L_k) = z_k + \epsilon_k$. Here ϵ_k is the error of sensor s_k. We assume the error$\{\epsilon_j\}_{\forall j \in \mathcal{S}_i}$ are independent and identically-distributed (i.i.d) according to a zero-mean Gaussian distribution with variance σ^2, *i.e.* $\epsilon_j \sim N(0, \sigma^2)$.



(a) Sensor readings are not linear to their locations. (b) Map sensor locations to another space, where sensor readings are linear to their locations.

Figure 1. An example of space mapping

To make this problem tractable, we need to assume a form of the function $\widetilde{f}_i(\mathbf{x})$. Work in [17] adopts a linear form to study this problem. Although linear functions are the simplest for this problem, they are not good candidates since the intensity of a field is generally non-linear in nature (and contour lines are generally not straight line segments). A better way is to use a combination of polynomial functions or Gaussian functions to represent $\widetilde{f}_i(\mathbf{x})$. This can be handled by kernel SVR [7]. We briefly present the approach as follows.

4.1.1 Theoretical Foundation

The idea of kernel SVR is employing space transform to map the original non-linear function $f(\mathbf{x})$ to another space where the function can be deemed as a linear function, which is then easy to be regressed. Figure 1 demonstrates a simple example of how space mapping works. In the network area, the sensor readings and their locations are not linearly-related. Whereas with a non-linear mapping y_1=x_1^2-$2x_1$ and y_2=x_2^2-$8x_2$+15, a location $\mathbf{x}$=(x_1, x_2) in the original space is then mapped to a location $\mathbf{y}$=(y_1, y_2) in another 2-dimensional space where z is then a linear function of $\mathbf{y}$, *i.e.*, z=y_1+y_2+2. The result can be projected back to the original space, *i.e.*, z=$(x_1$-$1)^2$+$(x_2$-$4)^2$.

Let us suppose Φ is an $\mathbb{R}^2 \mapsto \mathbb{R}^d$ mapping that maps a point $\mathbf{x}$ in the network area ϕ to a point $\mathbf{x}$ in a d-dimensional space $\mathbb{R}^d$. Consider z as a linear function of $\mathbf{y} = \Phi(\mathbf{x})$, *i.e.*,

$$z = w \cdot \mathbf{y} + b = w \cdot \Phi(\mathbf{x}) + b, \tag{2}$$

where $w \in \mathbb{R}^d$, $b \in \mathbb{R}$, and $\cdot$ denotes the inner product of two vectors. Then the canonical SVR formulation of this problem is:

$$\begin{aligned} &\underset{w,b}{\text{minimize}} && \tfrac{1}{2}||w||^2, \\ &\text{subject to} && \begin{cases} z_k - (w \cdot \Phi(L_k) + b) < \epsilon_k \\ (w \cdot \Phi(L_k) + b) - z_k < \epsilon_k \ . \end{cases} \end{aligned} \tag{3}$$

The idea of this formulation is to regress the field intensity as a linear function of node locations in the new space $\mathbb{R}^d$.

This problem can be worked out by solving its dual problem with a quadratic programming formulation. It turns out that only the inner product $K(\mathbf{x}_1, \mathbf{x}_2) = \Phi(\mathbf{x}_1) \cdot \Phi(\mathbf{x}_2)$ in $\mathbb{R}^d$, but not $\Phi(\mathbf{x})$, is involved in the optimization procedure and the result [24]. This greatly simplifies the optimization process as we do not need to know what the mapping actually is. $K(\mathbf{x}_1, \mathbf{x}_2)$ is called a *kernel*. With such a kernel SVR approach, $\widetilde{f}_i(\mathbf{x})$ can be represented by a set of locations $\{L_m\}_{\forall m \in \mathcal{SV}_i}$ (called *support vectors*) and their weights $\{\alpha_m\}_{\forall m \in \mathcal{SV}_i}$, where $\mathcal{SV}_i$ is a subset of $\mathcal{W}_i$:

$$\widetilde{f}_i(\mathbf{x}) = \sum_{m \in \mathcal{SV}_i} z_m \alpha_m K(L_m, \mathbf{x}) - b. \quad (4)$$

In general, kernel SVR is to find out a subset of node locations (*i.e.*, support vectors) and node readings that best represent the field intensity function with a kernel function.

4.1.2 Why Using Kernel SVR

Although SVR is proven effective in many disciplines (*e.g.*, [19, 21]), OACS is the first work that adopts kernel SVR for contour mapping in WSNs. In this section, we comprehensively justify the benefits to employ kernel SVR in providing a contour service.

Let us first illustrate the superiority of kernel SVR comparing with the scheme adopted in CME [30], a recent approach for WSN contour mapping which also suggests using similar machine learning approaches. CME formulates the problem of constructing a segment of contour line with value cv as a classification problem. The nodes with readings larger than cv are in class A, while the nodes with reading less than cv in class B. Then a classification boundary can be calculated by machine learning techniques and thus deemed as the contour line segment. Although such an approach is similar to ours since non-linear machine learning techniques are adopted, it relies simply on the shape of the distribution of the nodes' locations, while disregarding the sensor readings in the same class. As a result, it cannot generate satisfying contour lines.

Figure 2 shows a typical result generated by CME. First, CME generates the same curve for the requests with values between 80.0 and 90.0. This is because in this example scenario class A and class B remain the same if the values of the requested contour line are between 80.0 and 90.0, and consequently, the classification boundaries (*i.e., the contour line*) calculated by CME are the same. Second, CME tends to generate a biased contour line. This is due to the even distribution of the nodes in class A and class B in the top-left part and the bottom-right part of the dotted rectangle, respectively. CME does not consider the values of sensor readings. As a result, the nodes in class B in

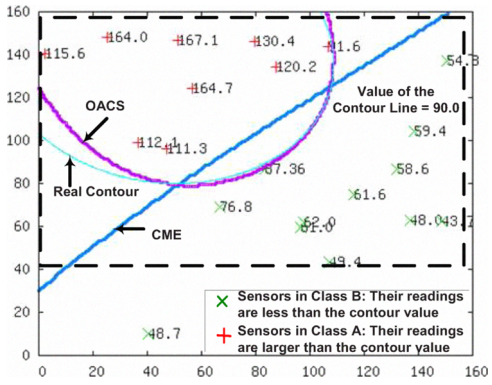


Figure 2. An example of the results generated by CME and OACS

the top-right and bottom-left corners, although with smaller readings, attract the classification boundary to go along top-right to bottom-left. OACS with kernel SVR which takes into account the sensor readings but not merely their locations, on the contrary, does not face these algorithmic inadequacies. Hence it can generate a more accurate result as demonstrated in the figure.

Besides its advantages comparing with the scheme adopted in CME, kernel SVR has many merits *per se*. First of all, it can equip OACS with a flexible method to deal with L-enquiries and M-enquiries. Examining its theoretical foundation, we can find that kernel SVR is a scheme that learns an output function based merely on its input samples, *i.e.*, $\mathcal{W}_i$. Therefore, by controlling the set of input samples, we can adapt it well to L-enquiries or M-enquiries. For L-enquiries, since we require $\widetilde{f}_i(x)$ to approximate $f(x)$ well in *only* the area around the contour line, $\mathcal{W}_i$ can be selected so that $\{z_k\}_{\forall k \in \mathcal{W}_i}$ are close to the contour value. Whereas for M-enquiries, many contour lines are required simultaneously and they are spreading throughout the cluster area; therfore, the regression result is important at anywhere of the network. Hence a larger $\mathcal{W}_i$ with node more evenly distributed can be the input for kernel SVR. Adaptively serving both types of enquires is a novel consideration of OACS in contrast to the current approaches which aim at either drawing a contour line efficiently or drawing a contour map efficiently, but not both.

Second, kernel SVR can conveniently handle the nonlinear nature of contour lines. Observe from Equation (4) that $\widetilde{f}_i(\mathbf{x})$ is a combination of kernel functions $K(L_m, \mathbf{x})$. If we want to represent the field as a combination of polynomial functions with degree d, we only need to assign a polynomial kernel $K(\mathbf{x}_1, \mathbf{x}_2)$=$(\mathbf{x}_1 \cdot \mathbf{x}_2 + 1)^d$ for the kernel-SVR. Similarly, if we want to represent the field as a combination of Gaussian functions, we just need a Radial Basis Function

(RBF) kernel with the form $K(\mathbf{x}_1, \mathbf{x}_2)$=$e^{-\gamma||\mathbf{x}_1-\mathbf{x}_2||^2}$, where e is the base of the natural logarithm. This provides a user the flexibility to select different non-linear curve functions according to the features of different sensing applications. Most importantly, the non-linearity of the results can thus be easily generated by applying the kernel method without causing much computational burden. The computational load of SVR is quite acceptable for a cluster head equipped with typical sensor platform such as Crossbow imote2 with a 400MHz MCU, since it just needs to solve a small-scale quadratic problem.

Finally, the result of kernel SVR is simple in representation. According to Equation 4, $\widetilde{f}_i(\mathbf{x})$ can be represented by a set of support vectors and their weights. A packet of a few tens of bytes is enough to encapsulate the result. OACS favors such a simple result since a cluster head hence only needs to send a single packet to the sink, which avoids high communication costs.

4.2. Precision Tuning via Active SVR-Based Approaches

Since there is no *a priori* knowledge on the requested contour line/map, initially, OACS lets λ percentage of nodes be on duty and then calculates the contour line/map based on these nodes. If the precision of the result does not match the user requirement (*i.e.*, λ<p), it will open a set of sleeping nodes and re-calculate the results based on the newly-added readings from this set of nodes.

Let $\mathcal{D}_i$ denote the set of sleeping nodes in cluster i, *i.e.*, $\mathcal{S}_i$=$\mathcal{D}_i \cup \mathcal{W}_i$. A straightforward way to select the set of to-be-opened nodes is to *randomly* select $(\lambda-p)n_i$ nodes from $\mathcal{D}_i$ and recalculate the contour line/map based on their readings. However, this scheme is inefficient. For example, for an L-enquiry, it may select the nodes far away from the actual contour line, but these nodes are almost useless. For an M-enquiry, it may select two nodes near each other, but they are redundant. To avoid such situations, the way to select these $(\lambda-p)n_i$ nodes should be carefully studied.

For classification problems in machine learning, a similar concern is how to select a set of training samples for enhancing the classification precision. Many active learning schemes have been proposed in recent approaches which actively select training samples for text [13, 28] or image classification problems [3]. Although these algorithms are specifically designed for classification problems, their underlying idea, *i.e.*, selecting the samples that can provide most information, sheds light on solving our problem. We hence tailor two active learning schemes for processing L-enquiries and M-enquiries to select $(\lambda-p)n_i$ nodes, as described in what follows.

4.2.1 L-enquiry Case

Consider the space $\mathbb{R}^d$ where the locations of nodes are projected by Φ. The contour line with value cv found by kernel SVR is hence an estimated hyperplane P_e in $\mathbb{R}^d$, where the points inside satisfy:

$$w \cdot \mathbf{y} + b = cv \Rightarrow w \cdot \Phi(\mathbf{x}) + b - cv = 0. \quad (5)$$

Since P_e is an estimation of the actual hyperplane P_a mapped from the actual contour line, it inevitably deviates from P_a. Let us consider a point $\mathbf{y}_1$ that is to one side of P_e and let d denote the distance from $\mathbf{y}_1$ to P_e. Note that the closer $\mathbf{y}_1$ is to P_e, the more likely $\mathbf{y}_1$ may actually be to the other side of P_a, *i.e*, the more uncertain for us to know whether $\mathbf{y}_1$ is actually on which side of P_a.

The distance d_j in $\mathbb{R}^d$ between a node s_j to the estimated hyperplane P_a is:

$$d_j = \left| \frac{w \cdot \Phi(L_j) + b - cv}{w} \right| \quad (6)$$

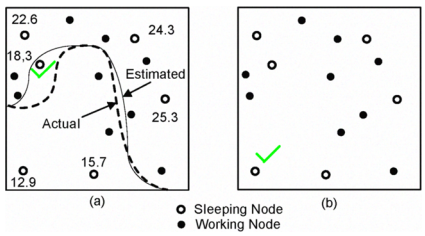
We then select a node $s_t \in \mathcal{D}_i$ as a to-be-opened node if it is the nearest sleeping node to the hyperplane since the uncertainty of whether the node is to the other side of the actual contour line is the maximum. In other words, its reading can potentially change the shape of the calculated contour line the most, and thus can best improve the accuracy of kernel SVR at the locations around the actual contour line. Formally,

$$\begin{aligned} t = & \operatorname*{argmin}_{\forall j \in \mathcal{D}_i} d_j = \operatorname*{argmin}_{\forall j \in \mathcal{D}_i} \left| \frac{w \cdot \Phi(L_j) + b - cv}{w} \right| \\ \Rightarrow & \quad t = \operatorname*{argmin}_{\forall j \in \mathcal{D}_i} |w \cdot \Phi(L_j) + b| \\ \Rightarrow & \quad t = \operatorname*{argmin}_{\forall j \in \mathcal{D}_i} \left| \widetilde{f}_i(L_j) \right| \end{aligned} \quad (7)$$

where the first transform is because $|w|$ and cv are the same for all nodes in $\mathcal{D}_i$ and the second is based on the mechanism of kernel SVR. See Figure 3(a) for an example where the value of the contour line requested is 20.0. The sleeping node, where the *estimated* field intensity 18.3 is closest to the contour value 20.0, should be opened to obtain its reading. This is because it is the most uncertain one regarding whether the node is to which side of the actual contour line.

Note that only $\widetilde{f}_i(\mathbf{x})$ is involved in solving Equation (7). This is very convenient since the $\widetilde{f}_i(\mathbf{x})$ in Equation (4) estimated by kernel SVR is already at hand based on the readings of $\mathcal{W}_i$.

Then s_t is turned on and added to $\mathcal{W}_i$. Kernel SVR runs again to update the contour line. The above procedure can be iterated until the precision of the generated contour line reaches the requirement, *i.e.*, pn_i nodes have been considered in producing the contour line.



(a) In processing an L-enquiry, the sleeping node, where the *estimated* field intensity 18.3 is the closest to the contour value 20.0, should be selected as a to-be-opened node. This is because it is the most uncertain regarding whether the node is to which side of the actual contour line. (b) In processing an M-enquiry, the sleeping node in the bottom-left corner is the farthest from the working nodes. It is the most uncertain one for the field intensity at that location. Hence it should be opened to obtain its reading.

Figure 3. Demonstration of the active node selection schemes.

4.2.2 M-enquiry Case

For processing an M-enquiry, the situation is a little bit different from how to process an L-enquiry discussed in Section 4.2.1. What we care now is to enhance the precision of kernel SVR everywhere in the cluster area. We discuss how to select a to-be-opened node as follows.

Consider again the space $\mathbb{R}^d$. The distance d_{kj} between two nodes s_k and s_j in $\mathbb{R}^d$ is:

$$d_{kj} = ||(\Phi(L_k) - \Phi(L_j)||^2 = \sqrt{K(L_k, L_k) + K(L_j, L_j) - 2K(L_k, L_j)} \quad (8)$$

Since $\widetilde{f}_i(\mathbf{x})$ is predicted by kernel SVR with the readings of the working nodes, the farther a location $\mathbf{x}$ from the working nodes is, the more uncertain the $\widetilde{f}_i(\mathbf{x})$ is. Therefore, to improve the accuracy of $\widetilde{f}_i(\mathbf{x})$, we should choose to open a sleeping node far away from the working nodes. We define the distance between a sleeping node to $\mathcal{W}_i$ as the minimum distance between it and all nodes in $\mathcal{W}_i$, *i.e.*, in $\mathbb{R}^d$ it is $\min_{\forall k \in \mathcal{W}_i} d_{kj}$. We select a sleeping node $s_t \in \mathcal{D}_i$ that has the maximum distance to $\mathcal{W}_i$. Formally,

$$t = \underset{\forall j \in \mathcal{D}_i}{\text{argmax}}(\min_{\forall k \in \mathcal{W}_i} d_{kj}) \quad (9)$$

Note that if we use an RBF kernel in SVR, $K(\mathbf{x}, \mathbf{x}) = e^{-\gamma||\mathbf{x}-\mathbf{x}||^2} = 1$. Equation (9) can be reduced to:

$$t = \underset{\forall j \in \mathcal{D}_i}{\text{argmax}}(\min_{\forall k \in \mathcal{W}_i} (-2K(L_k, L_j))) = \underset{\forall j \in \mathcal{D}_i}{\text{argmax}}(\min_{\forall k \in \mathcal{W}_i} ||L_k - L_j||), \quad (10)$$

which is exactly the same as selecting a sleeping node that has the maximum distance to $\mathcal{W}_i$ in the original space $\mathbb{R}^2$. See Figure 3(b) for an example where the sleeping node in the bottom-left corner is the farthest from the working nodes. It is the most uncertain one for the field intensity at that location. Hence it should be opened to obtain its reading.

5. Performance Study

To study the effectiveness of OACS in addressing the WSN contour mapping problem, we simulate a WSN. Without loss of generality, clusters are formed by evenly dividing the network into grids and their heads are randomly selected, which is similar to the scheme in [30]. The grid numbers are also selected based on the scheme in [30]. A widely-adopted SVR solver in SVM$^{\text{light}}$ [14] is employed to solve our kernel SVR problems for cluster heads[2].

Table 1. Simulation Settings

Area of sensor field	320m × 320m
Rode deployment scheme	Randomly deployed in a uniform manner
Communication range R_c	30m
Decay factor α and k	3 and 0.01
Packet size	48 bytes

The field intensity is generated by the model adopted in [18, 30], which is briefed as follows. M sources are randomly distributed in the network area, the locations of which are denoted by $\mathcal{T}=\{T_m\}_{m=1}^{M}$. Then the field intensity $f(\mathbf{x})$ at location $\mathbf{x}$ is determined by the summation of the diffusion from the sources:

$$f(\mathbf{x}) = \sum_{T_m \in \mathcal{T}} \frac{1}{(k||\mathbf{x} - T_m|| + 1)^{\alpha}}. \quad (11)$$

Note that this is a realistic model since in general the field is a cumulative result by several sources (*e.g.*, those that emit heat) while the effect of a source decays exponentially with a factor larger than 2 in space. The reading of a sensor s_i is then given by $f(L_i) + \epsilon$.

The details of our network settings are shown in Table 1, which is a typical WSN setting similar to that adopted in [30]. Accuracy of a contour line/map is calculated based on the MSE of the results with respect to the actual line/map. In our following performance study, for each setting we adopt different random seeds in every run and the results are averaged.

[2]The algorithm is implemented in C. It can easily be ported to current platforms of cluster heads, *e.g.* LiteOS [2].

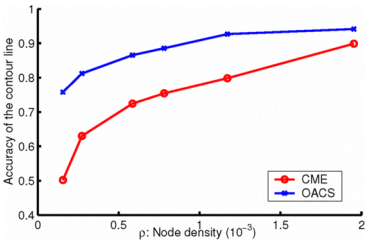


Figure 4. Accuracy comparisons between OACS and CME in processing L-enquiries

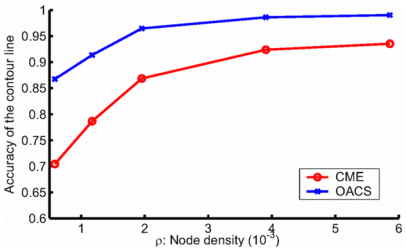


Figure 5. Accuracy comparisons between OACS and CME in processing M-enquiries

5.1. The Advantages of Kernel SVR

We first study the accuracy of our OACS in processing the L-enquiries, comparing with the recent CME approach proposed in [30]. For each L-enquiry $[cv, p]$, cv is a randomly-selected field intensity value. We set p=100% for fairness consideration as in this case OACS would not take any advantages of its active node-selection schemes. Our purpose is to compare the performance of our kernel-SVR algorithm and the classification algorithm adopted in CME.

We change the node density in processing L-enquiries. Figure 4 demonstrates an example result where M=3. We can see that OACS greatly outperforms CME especially when the node density is low. This is not surprising: CME relies only on the node-location distribution. It can generate accurate results only if the node density is very high. In contrast, OACS considers sensor readings in its kernel-SVR algorithm, which adapts well to low node density.

Note that the energy required for processing an L-enquiry for both CME and OACS is almost the same in this experiment (where the precision requirement p=100%). This is because both approaches let the nodes with readings close to the contour value report to their cluster heads.

We then compare OACS and CME in handling M-enquiries. Similar to the previous experiment, we also set p=100%. We change the node density and study the accuracy of the maps generated by OACS and CME. The accuracy results are demonstrated in Figure 5 where we randomly place six sources (*i.e.*,M=6) in the network. It shows that OACS also outperforms CME. The total number of packets that need to be sent by all in-network nodes are compared in Figure 6. We can see that if we need to generate a contour map with more contour lines, CME spends more energy than OACS does since CME has to calculate each contour line separately. In contrast, the number of packets remains constant for OACS since it generates the same result for an M-enquiry with the same p.

These simulations show that OACS outperforms CME in processing both L-enquiries and M-enquires in terms of accuracy of the results, while OACS is more energy-efficient

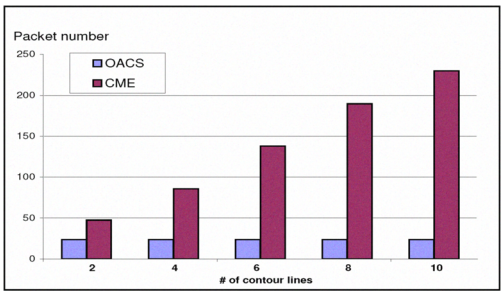


Figure 6. Total packet numbers sent by OACS and CME in processing M-enquiries

when calculating the contour map. Note that in these simulations, we set p=100%. In case that the user does not need such a high p, obviously OACS can conserve much more energy since it has a scheme to let some nodes sleep. CME, however, lacks such a scheme and as a result all nodes have to be in working mode. The energy consumption results are hence omitted when p<100%. In conclusion, we verify the advantages of employing kernel SVR in contour mapping.

5.2. The Advantages of Active Node Selection Scheme

Now we investigate how well our active node selection scheme performs in handling different user requirements on contour line/map precisions. We randomly place six sources (*i.e.*,M=6) and deploy 400 nodes in the network area. Two approaches are adopted: One is OACS with our active node selection scheme, while the other is another OACS version without such a scheme in which pn_i nodes are selected *randomly* from all the in-network nodes. We set λ=12% for OACS, *i.e.*, initially, 12% of in-network nodes are on-duty.

We study the accuracy of the results on L-enquiries and M-enquiries, in which we change the precision requirement p from 12% to 40%. Figures 7 and 8 show the results. We can see that OACS with an active node scheme per-

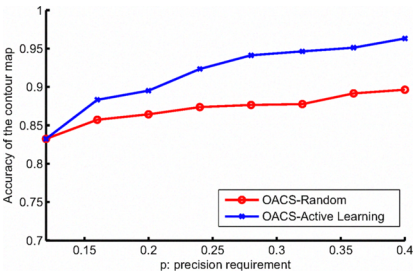


Figure 7. Accuracy as a function of precision requirement: L-enquiry case

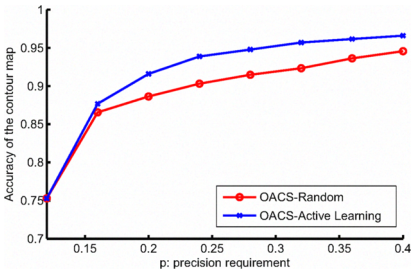


Figure 8. Accuracy as a function of precision requirement: M-enquiry case

forms much better than that without such a scheme[3]. This is also a natural result since our active node selection scheme aims at minimizing the uncertainty of the generated contour lines/maps. As a result, it can select the best set of to-be-opened sleeping nodes.

With the same setting, we further vary the node density and study how OACS with an active node selection scheme performs in processing M-enquiries. Figure 9 demonstrates the results, which show that the higher the node density, the more accurate the results. This is because when the node density is high, more candidates are available, and consequently a better set of to-be-opened nodes can be selected. Hence, the accuracy increases as the node density increases.

Finally, to study how OACS performs when the field intensity becomes more irregularly-distributed, we deploy 400 nodes in the network area and increase the number of sources M. Let OACS perform M-enquires with p=32%. Figure 10 shows that the accuracy of OACS almost remains unchanged when the source number increases. This demonstrates that the performance of OACS does not vary with how the field intensity is distributed, which verifies the nice adaptivity attribute of OACS.

[3]Note that the accuracy results of both schemes when p=12% are the same as in this case p=λ, which means that no sleeping nodes need to be turned on. As a result, the node selection schemes do not have any effects.

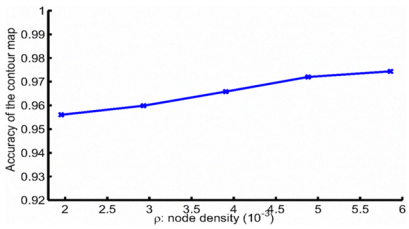


Figure 9. Accuracy as a function of node density

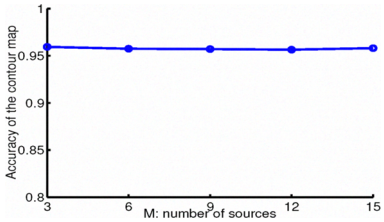


Figure 10. Accuracy as a function of source numbers

6. Conclusion and Future Work

In this work, we presented OACS as a promising contour mapping tool in handling diverse user requests. We showed that OACS can handle requests for both contour line and contour map energy-efficiently. Most importantly, it is intelligent in energy saving when providing users the flexibility to request contour lines/maps with different precision.

Our work demonstrated that the active learning and kernel SVR techniques from the machine learning field can be powerful tools for the contour mapping service in WSNs. There are, however, many remaining issues to be explored. We are particularly interested in developing on-line algorithms for contour mapping, where a contour line/map can be incrementally polished based on new sensor readings instead of re-running kernel SVR. We also intend to investigate how OACS performs if we adopt a batch mode active learning scheme [13] where all the to-be-opened nodes are selected simultaneously, instead of selecting the to-be-opened node one by one. Finally, how to calculate time-variant contour line/map is also of interest: A powerful contour service that can well exploit the temporal-correlations of sensor readings is still at large.

Acknowledgements

This work was substantially supported by a grant from the Research Grants Council of the Hong Kong Special Ad-

ministrative Region, China (Project No. CUHK4158/08E). J. Liu's work was supported in part by a Canadian NSERC Discovery Grant and an NSERC Strategic Project Grant.

References

[1] A. Ailamaki, C. Faloutos, P. S. Fischbeck, M. J. Small, and J. VanBriesen. An environmental sensor network to determine drinking water quality and security. *Sigmod Rec.*, 32(4):47–52, 2003.

[2] Q. Cao, T. Abdelzaher, J. Stankovic, and T. He. The liteos operating system: Towards unix-like abstractions for wireless sensor networks. In *Proc. of IPSN*, 2008.

[3] E. Chang, S. Tong, K. Goh, and C.-W. Chang. Support vector machine concept-dependent active learning for image retrieval. *IEEE Transactions on Multimedia*, 2, 2005.

[4] W.-P. Chen, J. C. Hou, and L. Sha. Dynamic clustering for acoustic target tracking in wireless sensor networks. *IEEE Transactions on Mobile Computing*, 3(3), 2004.

[5] Crossbow Technology. Imote2: High-performance wireless sensor network node. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/Imote2_Datasheet.pdf.

[6] J. C. Davis. Contouring algorithms. In *Proc. of the International Symposium on Computer-Assisted Cartography*, pages 352–359, Reston, VA, December 1975.

[7] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. In *Proc. of NIPS*, pages 155–161, 1996.

[8] G. Edmondo. Digital geologic field mapping using arcpad. In *Proc. of the Digital Mapping Techniques Workshop, U.S. Geological Survey Open-File Report 02-370*, 2002.

[9] S. Gandhi, J. Hershberger, M. Graphics, and S. Suri. Approximate isocontours and spatial summaries for sensor networks. In *Proc. of IPSN*, 2007.

[10] L. Gu and J. A. Stankovic. Radio-triggered wake-up for wireless sensor networks. *Kluwer Journal of Real-Time Systems*, 29(2-3):157–182, March 2005.

[11] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden. Distributed regression: an efficient framework for modeling sensor network data. In *Proc. of IPSN*, 2004.

[12] J. Hellerstein, W. Hong, S. Madden, and K. Stanek. Beyond average: Towards sophisticated sensing with queries. In *Proc. of IPSN*, Washington D.C., November 2005.

[13] S. C.-H. Hoi, R. Jin, and M. R. Lyu. Large-scale text categorization by batch mode active learning. In *Proc. of the 15th International World Wide Web conference (WWW)*, Edinburgh, England, UK, 2006.

[14] T. Joachims. Making large-scale svm learning practical. In *Advances in Kernel Methods - Support Vector Learning (eds. B. Schölkopf, C. Burges and A. Smola)*, MIT-Press, 1999.

[15] D. Krige. *A statistical approach to some mine valuations and allied problems at the Witwatersrand*. Master's thesis of the University of Witwatersrand, 1951.

[16] H. Liu, X. Jia, P. Wan, C.-W. Yi, S. K. Makki, and N. Pissinou. Maximizing lifetime of sensor surveillance systems. *IEEE/ACM Transactions on Networking*, 15(2):334–345, April 2007.

[17] Y. Liu and M. Li. Iso-map: Energy-efficient contour mapping in wireless sensor networks. In *Proc. of the IEEE ICDCS*, pages 36–44, Toronto, Canada, June 2007.

[18] X. Meng, T. Nandagopal, L. Li, and S. Lu. Contour maps: Monitoring and diagnosis in sensor networks. *Computer Networks*, 50(15):2820–2838, 2006.

[19] K. S. Ni and T. Q. Nguyen. Image superresolution using support vector regression. *IEEE Transactions on Image Processing*, 16(6):1596–1610, June 2007.

[20] R. Nowak and U. Mitra. Boundary estimation in sensor networks: Theory and methods. In *Proc. of IPSN*, pages 80–95, Palo Alto, CA, April 2003.

[21] J. Qiu, W. Sheffler, D. Baker, and W. S. Noble. Ranking predicted protein structures with support vector regression. *Proteins: Stucture, Function, and Bioinformatics*, 71(3):1175–1182, 2007.

[22] L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, Y. Wu, W. Kang, J. Stankovic, D. Young, and J. Porter. Luster: Wireless sensor network for environmental research. In *Proc. of the ACM Sensys*, November 2007.

[23] A. Silberstein, R. Braynard, and J. Yang. Constraint chaining: On energyefficient continuous monitoring in sensor networks. In *Proc. of the ACM SIGMOD*, Chicago, IL, 2006.

[24] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

[25] I. Solis and K. Obraczka. Efficient continuous mapping in sensor networks using isolines. In *Proc. of MobiQuitous*, San Diego, CA, July 2005.

[26] S. Srinivasan, K. Ramamritham, and P. Kulkarni. ACE in the hole: Adaptive contour estimation using collaborating mobile sensors. In *Proc. of IPSN*, 2008.

[27] TinyOS Community Forum. Tinyos: An open-source os for the networked sensor regime. http://www.tinyos.net.

[28] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.

[29] K. Whitehouse. The design of calamari: an ad-hoc localization system for sensor networks. M.Sc. Thesis, UC Berkeley, 2002.

[30] Y. Xu, W.-C. Lee, and G. Mitchell. CME: a contour mapping engine in wireless sensor networks. In *Proc. of ICDCS*, pages 133–140, 2008.

[31] W. Xue, Q. Luo, L. Chen, and Y. Liu. Contour map matching for event detection in sensor networks. In *Proc. of the ACM SIGMOD*, Chicago, IL, 2006.

[32] O. Younis and S. Fahmy. HEED: a hybrid, energy-efficient. distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing*, 3(4):336–379, 2004.

[33] J. Zhao, R. Govindan, and D. Estrin. Residual energy scans for monitoring wireless sensor networks. In *Proc. of the IEEE Wireless Communications and Networking Conference*, pages 17–22, March 2002.

[34] Y. Zhou, M. R. Lyu, and J. Liu. On sensor network reconfiguration for downtime-free system migration. *ACM/Springer MONET*, 14(2):241–252, April 2009.