Towards Spatial Window Queries Over Continuous Phenomena in Sensor Networks

Guang Jin and Silvia Nittel

Department Of Spatial Information And Engineering

University of Maine, USA

{jin,nittel}@spatial.maine.edu

Abstract

Recent research on sensor networks has focused on efficient processing of declarative SQL queries over sensor nodes. Users are often interested in querying an underlying continuous phenomenon, such as a toxic plume, while only discrete readings of sensor nodes are available. Therefore, additional information estimation methods are necessary to process the sensor readings to generate the required query results. Most estimation methods are computationally intensive, even when computed in a traditional centralized setting. Furthermore, energy and communication constraints of sensor networks challenge the efficient application of established estimation methods in sensor networks. In this paper, we present an approach using Gaussian Kernel estimation to process spatial window queries over continuous phenomena in sensor networks. The key contribution of our approach is using a small number of Hermite coefficients to approximate the Gaussian kernel function for sub-clustered sensor nodes. As a result, our algorithm reduces the size of messages transmitted in the network by logarithmic order, thus, saving resources while still providing high quality query results.

I. Introduction

Today, micro-scale sensing in combination with tiny computing and communication devices form sensor networks which enable us to measure physical environmental phenomena in a level of detail that we were not able to observe and measure before.

A. Problem Definition

Sensor networks continuously collect information about the physical environment. Due to their massively parallel, distributed, failure-prone and energy-constrained nature, sensor networks are tedious to program. The database community takes the standpoint that viewing a sensor network as a distributed database system, which accepts and processes declarative SQL queries, significantly simplifies the programming and deployment task of sensor networks. Approaches such as TinyDB[1], [2] and Cougar[3], [4] are the first implementations of such small-scale and distributed DBMSs.

Environmental monitoring is usually interested in spatially continuous phenomena, such as microclimates around redwood trees[5] or within vineyards [6]. Spatial window queries are important for users to understand the underlying phenomena, while only sensor readings at discrete and limited node locations are available. Although many available techniques are able to estimate a continuous phenomenon based on point samples, the current research challenge is finding a resource-efficient adaption of those methods for the resource constrained environment of sensor networks.

B. Our Contribution

In this paper, we present SWOP(*Spatial Window Query Over Phenomena*), an efficient algorithm to support in-network spatial window query processing based on Gaussian Kernel estimation. In general, SWOP first groups sensor nodes into sub-clusters according to node locations. Next, SWOP transforms Gaussian weighted readings into a Hermite series for each cluster representing the detailed information about local sensor nodes to minimize the communication cost. In this way, a large set of node readings can be represented by only a small number of Hermite coefficient terms, while the communication cost for node IDs and individual sensor readings is reduced. Therefore, the total amount of data transmitted inside the network is reduced by logarithmic order, and the

computation cost on individual nodes is kept constant. A centralized computer or a microserver deployed in the network with more powerful resources evaluates the transformed, minimized data set to generate the final spatial window query result. Due to the fast convergence speed of Hermite expansion, the difference between results of SWOP and the traditional centralized Gaussian Kernel estimation is minimal. The experimental results confirm our expectation and demonstrate the high performance of SWOP for different data sets.

The remainder of this paper is organized as follows: The preliminaries of SWOP are formulated in Section II. Section III reviews several estimation techniques, and Section IV proposes the theoretical foundation of SWOP. We describe the algorithms of SWOP and analyze its computation and communication cost in Section V. The experimental results are illustrated in Section VI. In Section VII, we explore the related work. We conclude and describe the future work in Section VIII.

II. Preliminary

A. Basic Concepts

In this section, we first formalize several relevant concepts for querying continuous phenomena using sensor networks.

Definition 1: A spatial continuous phenomenon is a spatial scalar field that represents the variation of a scalar property over a geographical space. Examples of the scalar properties are temperature, wind-speed, or the concentration of a gas pollutant in the air. Formally, given a geographical space S and a class of scalar values V, a *spatial continuous phenomenon* is a function F whose domain is S and codomain is V [7].

Definition 2: A monitored region, R, is a subarea of the geographical space and is observed by a particular sensor network. Although the physical world is a 3D Euclidean space, in this paper we mainly assume a 2D Euclidean space.

Definition 3: For each point p in R the phenomenon value is represented by $Y(p)$. Additionally, we use s_i to identify an individual sensor node and its spatial location. $Y(s_i)$ is defined as the sensor reading and phenomenon value at the sensor node s_i. For an estimation result for any point p in R, we use $\hat{Y}(p)$.

B. Modeling Sensor Networks And Spatial Window Queries Over Phenomena

A sensor database system is a "mediator" between users and the underlying phenomena, as shown by Fig.1. Most of the current research work focuses on in-network processing
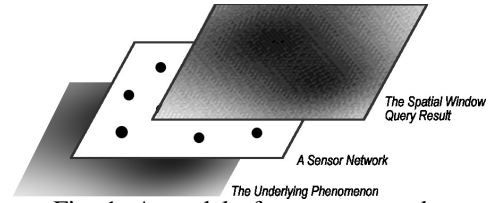

Fig. 1: A model of sensor networks

figure

of temporal queries. Local processing of temporal queries at individual nodes benefits from the minimized communication messages[8], [9]. Spatial queries, especially over an underlying continuous phenomenon, are not well supported today, since in-network processing of spatial queries is much more complex and expensive than processing temporal queries, and requires the inevitable communication overhead among neighboring nodes.

Spatial queries can be roughly categorized into two groups based on the spatial predicates in the declarative SQL. One type is *point queries* which return values for particular points in space. Here, the query result can be a sensor reading at a well-defined sensor node location, or an estimation result for a point in-between sensor nodes based on neighboring sensor readings. Another type of spatial queries is *spatial region queries* which return values for a continuous spatial region, R. Different from point queries at well-defined node locations, point queries in-between sensor nodes and spatial range queries require additional estimation techniques. For example, Voronoi diagram and TIN based approaches were used to support spatial aggregation queries[10]. Processing estimation techniques, however, requires additional resources from networks.

An important class of sensor network applications is the observation of spatially continuous phenomena such as micro-climates or the distribution of gas pollutants. In this case, simple aggregated statistical information is not sufficient. Here, a special type of spatial region queries becomes relevant, i.e. *spatial window queries* which return the phenomenon's continuous distribution within a well-defined boundary of the monitored region R, e.g."SELECT p.temperatureField FROM Phenomenon p WHERE p.location WITHIN region R". On the contrary to spatial aggregation queries, a spatial window query returns values at arbitrary points in R as shown by Fig.1 and presents the result in a digital format, such as a digital image in a user-defined resolution. The number of points in the result is usually more than the number of sensor nodes within the spatial window predicate. Additional estimation techniques are necessary to "fill" the blank points in-between sensor nodes.

In traditional settings, spatial window queries were processed in a centralized setting based on all raw readings, while constrained sensor networks favor in-network processing of estimation techniques to minimize the data

communication cost among nodes.

III. Answering Spatial Window Queries Using Estimation Techniques

Before applying estimation techniques in sensor networks, we need to answer several questions, such as "how to represent a spatial window query result" and "how to process additional estimation techniques in the network".

A. Voronoi Diagram

Voronoi diagrams provide a simple approximation model, where the monitored region, R, is partitioned into a set of "Voronoi cells" based on the locations of sensor nodes. The phenomenon in a cell is presented by the sensor reading at the cell center. For example, a spatial aggregation query result can be represented as a weighted summary of sensor readings according to the sizes of Voronoi cells[10]. To answer a spatial window query, a sensor network needs to find the Voronoi diagram intersected with the query predicate as a set of edges. Suppose n ($n > 3$) sensor nodes involved, the upper bound of the number of edges is $3n - 6$, and the lower bound is $n - 1$ [11]. A distributed algorithm can find the Voronoi edges, but requires additional resources from the network [10]. Similar neighboring sensor readings can be merged [12], but such compression ideas are also applicable to raw sensor readings. Based on the Voronoi diagram, we can generate a TIN(Triangulated Irregular Network)[10] as a 3D terrain about the underlying phenomenon. TINs are useful to generate contour maps about the phenomena, but consume similar resources as Voronoi diagrams do. Overall, using Voronoi diagram based approaches needs an expensive algorithm to find and represent the Voronoi cells, and the estimation results are rather coarse.

B. Spatial Regression

Spatial regression attempts to represent the underlying phenomenon as an equation. The solution of regression can be represented as, $\hat{Y}(q) = \sum_{i=1}^{k} [w_i \cdot f_i(q)]$, where the estimated value for any point in R is a weighted summary of predefined basis functions, $f()$s. In a 2D polynomial regression[13], similar to a 1D polynomial temporal regression[8], $f()$s are polynomial functions of x and y coordinates. For instance, an underlying phenomenon can be represented by a quadratic polynomial function, $\hat{Y}(q) = w_0 + w_1 \cdot x_q + w_2 \cdot y_q + w_3 \cdot x_q^2 + w_4 \cdot y_q^2 + w_5 \cdot x_q y_q$, if x_q and y_q represent the x and y coordinates of point q. Another form of spatial regression is known as Kernel regression, where the basis functions are a set of kernel

functions, $k()$s[14]. Each kernel function has a unique kernel center in space, and is typically a predefined non-increasing function of the Euclidian distance between the center and the input point. Given k basis functions $f()$s and n sensor readings, to minimize the MSE(Mean Squared Error), the weights for basis functions are computed by, $W = (\sum_{i=1}^{n} (F(s_i)^T F(s_i)))^{-1} \sum_{i=1}^{n} (F(s_i)^T Y(s_i))$, where W and $F()$ are the vector format of w and $f()$, i.e. $W = [w_1, w_2, \cdots, w_k]^T$, and $F(s_i) = [f_1(s_i), f_2(s_i), \cdots, f_k(s_i)]$.

To represent the estimation result, a sensor network only needs to return the estimated weights, W, which require minimal resources from a sensor network compared with transmitting raw sensor readings. One way to find the estimated W is to aggregate the two matrixes, $\sum_{i=1}^{n} (F(s_i)^T F(s_i))$, and $\sum_{i=1}^{n} (F^T Y(s_i))$ within the network, and compute W outside. For a more heterogenous spatial phenomenon, more basis functions are required to increase the estimation quality. However, the cost on networks increases exponentially when applying more basis functions, since each node needs $k^2 + k$ data to aggregate for k basis functions. Choosing special forms of kernels, such as Block kernel or Cone kernel[14], in a kernel regression may relax the data requirement for individual nodes, but the quality of estimation result is deteriorated due to the discontinuity of the kernel functions. Another possible solution is using the distributed matrix inversion operations[14], [15] by exchanging local aggregated matrices among neighboring nodes. This solution is still expensive with regard to the communication cost, since distribution matrix inversion algorithms require more than one iteration of exchanging local information to achieve a satisfactory error tolerance. Although the W requires minimal resources, the communication cost to find W deteriorates the performance of constrained sensor networks. If a network is monitoring a dynamic phenomenon in a frequent temporal rate, the distributed matrix inversion operations face more difficulties.

For sensor database systems, efficient processing of spatial window queries requires the relaxation of the cost of in-network estimation processing and the in-network representation of the query result. The two requirements are often intertwined. For example, a spatial regression uses minimal resources to represent the result W, but requires iterations of communication to find the W. We need to find an estimation model to minimize both requirements and still maintain the high quality of spatial window query results.

IV. Theoretical Foundation of SWOP

A. Kernel Estimation

SWOP is based on another well-known estimation model, Kernel estimation. Different from Kernel regression [14], Kernel estimation is a non-parametric estimation and can be stated as "total amount of observed values per unit area". Kernel estimation is also a special spatial moving-average method, so the estimation result is robust against noise. For a point q in monitoring region R, Kernel estimation estimates the phenomenon value at that point as,

$$\hat{Y}(q) = \frac{1}{\tau^2} \sum_{i=1}^{n} Y(s_i) K\left(\frac{|s_i - q|}{\tau}\right), \text{where } s, q \in R. \quad (1)$$

In Eq.1, $Y(s_i)$ represents a reading for the sensor node s_i, $\hat{Y}(q)$ is the estimation result for the point q, and $|s_i - q|$ presents the Euclidian distance between the point q and the sensor node s_i.

In a simple distributed algorithm[16] similar to [2], [13], every sensor node evaluates its neighboring points in R. In a routing tree based protocol, each node aggregates its partial result with the partial results from its children[2], [13]. The size of total data extracted from the network is linearly scaled by the number of points to represent the Kernel estimation result. To answer a high resolution spatial query result, we need more estimation points than the number of involved sensor nodes. Therefore, a simple distributed solution often makes no significant improvement compared with a traditional centralized solution, especially when raw readings are compressed in the network.

B. Gaussian Kernel and Fast Transforms

The main difficulty of applying Kernel estimation in sensor networks is the entangled links between estimation points and sensor readings. Neither direct evaluating sensor readings outside the network nor direct evaluating estimation points within the network is resource-efficient.

SWOP chooses another way to efficiently process the Kernel estimation based on the Gaussian kernel. Gaussian kernel is a very smooth kernel function. Gaussian Kernel estimation has a wide range of applications, such as financial analysis[17] and image processing[18], and estimates the phenomenon value at the point q as,

$$\hat{Y}(q) = \frac{1}{\tau^2} \sum_{i=1}^{n} Y(s_i) e^{-|s_i - q|^2/\tau^2}, \text{where } s, q \in R. \quad (2)$$

To break the entangled links between estimation points and sensor nodes, SWOP has to transform the Gaussian kernel. Two fast transforms are available, the FGT(Fast Gaussian Transform)[19] and the IFGT(Improved Fast Gaussian Transform). Both fast transforms use an infinite

series to approximate the Gaussian kernel and truncate insignificant series terms to accelerate the evaluating speed. This feature is very favorable to SWOP, since the truncated series can use a small size of data to represent detailed information about all raw readings. To achieve more efficient processing in constrained sensor networks, SWOP needs to choose an appropriate transform which minimizes the data requirement of communication.

Let's first explain two transforms in the 1D space. The FGT utilizes the Hermite expansion to represent the exponential function as,

$$e^{-|s_i - q|^2/\tau^2} = \sum_{j=0}^{\infty} \frac{1}{j!} \left(\frac{\Delta s_i}{\tau}\right)^j h_j\left(\frac{\Delta q}{\tau}\right), \quad (3)$$

where $\Delta s_i = s_i - s_*$, $\Delta q = q - s_*$ and the Hermite functions $h_j(x)$ are defined by

$$h_j(x) = (-1)^j \frac{d^j}{dx^j}\left(e^{-x^2}\right).$$

The FGT needs to group sensor nodes into sub-clusters. Here, s_* is the cluster center which satisfies $|s_i - s_*|/\tau < 1$, so the Hermite coefficients converge to zero and the Gaussian kernel can be safely approximated by the first p terms,

$$\hat{Y}(q) \approx \frac{1}{\tau^2} \sum_{j=0}^{p} A_j(s) h_j\left(\frac{\Delta q}{\tau}\right), \quad (4)$$

where the Hermite coefficients $A_j(s)$ are defined as

$$A_j(s) = \frac{1}{j!} \sum_{i=1}^{n} Y(s_i) \left(\frac{\Delta s_i}{\tau}\right)^j. \quad (5)$$

The IFGT(Improved Fast Gaussian transform)[18] factorizes the Gaussian kernel as

$$e^{-|s_i - q|^2/\tau^2} = e^{-\frac{\Delta s_i^2}{\tau^2}} e^{-\frac{\Delta q^2}{\tau^2}} e^{-\frac{2\Delta s_i \Delta q}{\tau^2}} \quad (6)$$

and uses Taylor expansion to approximate,

$$e^{-2\Delta s_i \Delta q/\tau^2} = \sum_{j=0}^{\infty} \frac{2^j}{j!} \left(\frac{\Delta s_i}{\tau}\right)^j \left(\frac{\Delta q}{\tau}\right)^j.$$

In IFGT, Eq.2 is approximated as

$$\hat{Y}(q) \approx \frac{1}{\tau^2} \sum_{j=0}^{p} C_j(s) e^{-\Delta q^2/\tau^2} \left(\frac{\Delta q}{\tau}\right)^j, \quad (7)$$

where the Taylor coefficients $C_j(s)$ are defined as

$$C_j(s) = \frac{2^j}{j!} \sum_{i=1}^{n} Y(s_i) e^{-\Delta s_i^2/\tau^2} \left(\frac{\Delta s_i}{\tau}\right)^j. \quad (8)$$

Here, the cluster center satisfies $2|\Delta s_i||\Delta q|/\tau^2 < 1$, so the Taylor coefficients converge to zero and terms after the first p terms can be safely truncated.

To safely truncate the series, both FGT and IFGT group the sensor nodes into sub-clusters with a radius
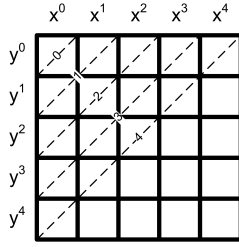
Fig. 2: Coefficient polynomial order

figure

smaller than the required bandwidth. For each cluster, the Hermite coefficients, Eq.5, and the Taylor coefficient, Eq.8, can represent the detailed sensor locations and readings, therefore are the only data needed from the network, as shown by Eq.4 and Eq.7 respectively. SWOP need to choose a transform which requires a smaller number of coefficient for the same quality criteria. If we assume $\rho_s = |s_i - s*|/\tau$ is the normalized cluster radius, and $\rho_p = |q - s_*|/\tau$ is the normalized distance between query points and cluster centers, the Hermite expansion requires $\rho_s < 1$ while the Taylor expansion requires $2\rho_s\rho_q < 1$. The Taylor expansion also requires $\rho_q > 1$, or the estimation result ignores some important readings outside the range[18]. Therefore, the Taylor expansion requires smaller cluster radiuses than the Hermite expansion does. Even if we set the cluster size equal in both transforms, the Hermite series converges faster than the Taylor series does, which is proven by the error bound of FGT [20] and IFGT [18]. Thus, SWOP chooses the Hermite expansion to transform raw sensor readings because of the faster convergence speed of Hermite series and the relaxed cluster size.

C. Data Reduction in High Dimension Space

In a high dimensional space, the FGT treats Eq.3 as a product of p-terms Hermite expansion along each dimension, and requires p^d terms in total for a d-dimensional space [19]. In the IFGT, the total number of terms in a d-dimensional space is $\binom{p+1}{d}$ by treating the vector product as a scalar dot-product in Eq.6 [19]. Differentiating different dimensions is important when a phenomenon is not isotropic, i.e. the phenomenon is directionally different, so the Kernel estimation can use different bandwidths for different dimensions. Consequently, the data size of Hermite expansion increases exponentially in a high dimensional space, while the data requirement of Taylor expansion is relaxed, and grows approximately linearly [18].

Unfortunately, in real applications, the location of sensor node is at least 2D. The constrained environment of sensor networks requires SWOP to relax the data requirement of Hermite expansion in a high dimensional space. Eq.4 and Eq.7 illustrate that the truncation of both expan-

sions benefits from the elimination of insignificant series terms with values close enough to 0. Eq.5 and Eq.8 explain that the significance of both expansion terms is determined by the polynomial order of the coefficients, since $\rho_s < 1$ and $2\rho_s\rho_q < 1$. In the 1D scenario, taking the first p terms means taking the expansion terms with polynomial order lower than $p - 1$. In a higher dimensional space, SWOP reorders the series terms based on their significance, i.e. the polynomial order. For example, in a 2D space, the original Hermite coefficients from the network can be represented as a 2D array as shown by Fig.2, where each element is a sum of products of x and y Hermite coefficients. To get the Hermite coefficients less than the quartic order, SWOP only requires the upper-left triangular matrix, since the lower-right triangular elements are much closer to 0 than the upper-left elements. In this way, SWOP relaxes the data requirement of Hermite expansion from p^d to $\binom{p+1}{d}$ based on $(p - 1)$ polynomial order in a d-dimensional space, and requires the same cost as the Taylor expansion does. Our experimental results confirm our expectation that by truncating Hermite series coefficients based on the polynomial order, SWOP outperforms the original FGT, since for the same data requirement, SWOP returns better estimation results than the original FGT does.

D. Processing in Mobile and Static Networks

SWOP needs to group sensor nodes into sub-clusters according to the node locations and transforms raw readings into the Hermite coefficients. For each sensor cluster, SWOP is a special aggregation query. For different types of networks, SWOP uses different processing strategies.

The main difference for processing SWOP in mobile and static networks is the clustering algorithm. The FGT does it by dividing the space into regular grid cells, named 'Boxes'[19]. This clustering algorithm is very simple, but may introduce many empty boxes due to the uneven distribution of sensor nodes, especially when nodes are mobile. The optimal clustering, however, is known to be NP hard [21]. Several sub-optimal clustering algorithms, such as K-means, G-means and hierarchical clustering[22], are still useful for static sensor networks. For a mobile sensor networks, directly distributed applications of such sub-optimal clustering algorithms are not efficient, since the computation and communication cost is too expensive for the constrained environment. Distributed clustering algorithms, such as HEED[23] and LEACH[24], provide solutions for mobile sensor networks. In distributed clustering algorithms, each sensor node can be a cluster head or belong to a cluster. A sensor node with more remaining energy and more potential communication links with others more likely announces itself to be a cluster head. Other nodes can join an appropriate cluster by detecting

and analyzing the cluster-head announcements. HEED has several advantages over LEACH, such as supporting multi-hop clustering and different clustering preferences. Thus, we choose HEED as the basic clustering method in SWOP for mobile networks. In SWOP, the cluster radius should be smaller than the Kernel bandwidth, τ, to satisfy the convergence condition, while bigger clusters are favorable to achieve a higher compression rate. Thus, we set the cluster radius to 0.9τ in SWOP. The only issue of applying distributed clustering algorithms is the required cluster size might be larger than the possible communication range of sensor nodes. In this case, SWOP allows small clusters to merge together till required cluster radius similar to what we did in [25].

After being clustered in SWOP, each sensor cluster is identified by its cluster center, s_*, which may not coincide with the location of the cluster head node. The detailed information about individual sensor readings and sensor locations can be transformed into a small number of Hermite coefficients. Because of the fast convergence speed of Hermite expansion, we expect the difference between the results of SWOP and centralized Kernel estimation is minimal. For each cluster in a mobile network, the Hermite coefficients are a special aggregation data and are routed to the central computer by the cluster heads [23]. In a static network, SWOP identifies clusters centrally and dispatches a set of multi-aggregation queries into the network over non-overlapped sub-clusters [26].

V. SWOP

A. Normalized Kernel

Although Eq.1 is useful in many cases, the normalized Kernel estimation model performs better when nodes are unevenly distributed. The normalized model estimates the phenomenon value at the point q as the estimation value by Eq.1 divided by total kernel weights,

$$\hat{Y}(q) = \frac{\sum\limits_{i=1}^{n} Y(s_i) e^{-|s_i - q|^2/\tau^2}}{\sum\limits_{i=1}^{n} e^{-|s_i - q|^2/\tau^2}}, \text{where } s, q \in R. \quad (9)$$

In Eq.9, τ^{-2} in the denominator and the numerator is canceled. Based on the normalized model, the estimation result of SWOP is robust against the uneven distribution of sensor nodes, e.g. lossy readings.

B. Description of SWOP Algorithm

After being awoken by a query, in SWOP, a sensor node can be either a cluster head or a non-head node belonging to a cluster based on the clustering algorithm. In a static sensor network, we can apply a centralized, more optimal and more expensive clustering procedure, while a distributed clustering algorithm is appropriate for a mobile sensor network. After identifying the cluster center, the cluster-head node transforms raw readings into the Hermite coefficients.

TABLE I: Algorithm on cluster heads

table
1. msg = receiveMsg();
2. if(!fromSameCluster(msg)){
3. if(countNumOfNodesInCluster() > tolerance){
4. numeratorSeries = HermiteExpand(msg, myLocation);
5. denominatorSeries = HermiteExpand(msg, myLocation, myReading);
6. myMsg = packMsg(numeratorSeries, denominatorSeries,clusterCenter);}
7. else myMsg = packMsg(msg,myReading, myLocation,clusterCenter);
8. routeToCentralBase(myMsg);}
9. else routeToCentralBase(myMsg,msg);

In clustering-based protocols, where a non-head node usually has a direct communication link with its cluster head, SWOP allows a non-head node to simply send its sensor reading and location (or its identity if the location information is cached) to its cluster head. A cluster head converts raw readings into Hermite coefficients for both denominator and numerator in Eq.9, if the number of nodes in cluster is large enough. If a cluster is small, the cluster head simply returns raw data including sensor readings and locations to the central computer as shown by Tab.I. In a static network, we treat SWOP as a set of multi-aggregation queries over non-overlapped clusters, so SWOP is efficiently processed by the method provided by [26].

TABLE II: Algorithm on the central computer

table
1. get_query(specifications);
2. send_to_sensors(init_msg);
3. do wait_for(responses);
4. till get_all();
5. generate_estimation_result();
6. return(estimation_result);

After receiving a spatial window query, a central computer first invokes the necessary sensor nodes and disseminates initial messages into the network as shown by Tab.II. After all cluster heads return their Hermite coefficients, the central computer needs to reconstruct the weighted readings based on Eq.3 from the partial expansion and sum them with uncompressed readings. After the central computer estimates all points within the spatial window predicate based on a user-defined resolution, a visual image is returned.

C. Analysis of SWOP

1) Computation Cost: The computation cost of the central computer is related to the number of points m for a user-defined resolution, the number of clusters k and the chosen polynomial order $p-1$. The computation complexity can be formulated as $O(m*k*\binom{p+1}{2})$. The chosen polynomial order $p-1$ and the number of clusters k are much smaller than the number of invoked sensor nodes n for an acceptable error-tolerance. The computation cost for a central computer can be relaxed as O(m) which is linearly relative to the user-defined resolution for a spatial window query. Further computation acceleration can be achieved by differentiating the Hermite series around the estimation points as shown by [19]. In this paper, we ignore it because the computation is done by a central computer or a microserver, and the computation cost has no effect on the network.

The computation cost of a sensor node is dominated by the clustering procedure, since aggregating Hermite coefficients requires a constant cost as shown by Eq.5 and Tab.I. In a mobile sensor network, SWOP chooses a distributed clustering algorithm which has to consume resources from the network. If SWOP uses a HEED-based clustering algorithm and the required cluster radius is smaller than the communication range, SWOP requires O(1) resources from the network, which has been proven by [23]. If the cluster radius is larger than the communication range and only merge operations among small clusters are allowed [25], in the worst case in which all nodes need to be merged into a single cluster, the complexity of the merge operation can be approximated as $O(log(n))$ [27] for n nodes. Thus, the total computation complexity of a sensor node is $O(log(n))$ in the worst case and $O(1)$ in general cases for a mobile network. For a static sensor network, the clustering procedure can be predetermined by the central computer, so the computation complexity can be further relaxed.

2) Communication Cost: The size of the total data extracted from the network of SWOP is determined by the number of clusters c and the chosen polynomial order $p-1$. For each cluster, $k_p+2\binom{p+1}{2}k_i$ bits are needed for the Hermite coefficients, where k_p is the required bit-length to represent a point for the cluster center s_* and k_i is the required bit-length to represent a term of the Hermite series. Whereas, $l(k_p+k_i)$ bits are needed for the raw data if l sensor nodes are in the cluster.

The total communication cost within the network depends on particular communication protocols and the network topology. For a mobile environment, clustering protocols are preferable and typically the non-head nodes and their cluster-heads have a direct communication link. Under the worst topology, in which all cluster-head nodes

form a line structure, receive and relay messages one by one, SWOP requires $0.5(1+c)c(k_p+2\binom{p+1}{2}k_i)$ data for the total communication between cluster head nodes within the network. If h represents the number of cluster heads before a particular cluster head on the line to the central base, the cluster head receives $(c-h-1)(k_p+2\binom{p+1}{2}k_i)$ and send $(c-h)(k_p+2\binom{p+1}{2}k_i)$ to the next hop. In the best case, in which every cluster head can directly send its messages to the cluster base, the total communication cost within the network is $c(k_p+2\binom{p+1}{2}k_i)$.

For a static environment using routing tree based protocols, SWOP is a set of multi-aggregation queries over non-overlapped spatial clusters. The in-network query processing can be optimized by the algorithm provided by [26], [28], and SWOP is categorized as a *min* query in [26]. Further cost evaluation of communication can be found in [26], [28].

D. Discussion Of SWOP

Most processing techniques only focus on the properties of the sensor readings or estimation results. SWOP, however, addresses the spatial properties of the sensor nodes. The compression gain of SWOP results from the spatial clustering. For instance, in a dense cluster with more than $[k_p+2\binom{p+1}{2}k_i]/(k_p+k_i)$ sensor nodes, the raw readings can be reduced to the first $p-1$ order Hermite coefficients in a 2D space. Since SWOP focuses on the spatial properties of the sensor nodes, the available compression techniques can also be applied on SWOP's transformed data among different clusters in the multi-hop transmission. Due to normalized Kernel estimation's robustness against noisy and lossy samples, SWOP performs well in the noisy and lossy environment of sensor networks. Furthermore, we do not limit SWOP to static sensor networks. The denominator in Eq.9 only presents the spatial properties of invoked sensor nodes. In a static sensor network, the spatial properties of sensor nodes are typically cached by the central computer, so more than half of the communication can be saved by excluding the denominator in normalized Kernel estimation from the in-network communication. A centralized clustering can also find better clustering patterns and therefore help SWOP to accomplish even a higher data compression rate.

By significantly compressing the 2D or 3D spatial readings in SWOP, we can treat the transformed data as a special snapshot in 1D temporal space. Therefore, any 1D temporal processing method, such as [8], [9], is applicable to the transformed SWOP data. The temporal processing of SWOP is another important issue; we do not discuss it in this paper due to the space limitation.

Overall, the computation complexity of SWOP with regard to distributed sensor nodes is constant and com-

munication messages are minimized. SWOP is practical for both static and mobile sensor networks.

VI. Experimental Evaluation

SWOP is a set of multi-aggregation queries over non-overlapped spatial clusters in routing-tree based sensor networks. We assume SWOP to be running in a more challenging environment, mobile sensor networks, so we use a x-y coordinate (128*bits*) to identify sensor nodes, choose the HEED-based clustering procedure, and assume the node communication range is bigger than the required cluster radius (i.e. a direct communication link between a non-head member and its cluster head). We implemented SWOP in Java and ran it over different data sets. In our simulations, the behavior of sensor networks is simulated by treating each sensor node as a thread running independently and communicating with each other by exchanging messages. The data sets consist of two real data sets from the CalCOFI survey off the coast of Southern California [29] and from a snapshot from an experiment in the Intel Lab[30], and two synthetic data sets. Without losing any generalization, we normalized the sensor readings to $[0, 1]$. How to find an optimal bandwidth has been researched well for Kernel estimation[31], and the fast optimization algorithm [32] for Gaussian kernel bandwidth is also available. Therefore, we only test SWOP under pre-chosen bandwidths. Especially, for two synthetic data sets we set the bandwidth fixed, since we want to test the SWOP estimation results with alternative estimation techniques by their processing costs based on different estimation qualities. Since many related works[14], [13], [10] only compare their approaches with respect to centralized solutions, it is hard to do cross-evaluation among them. In our experiments, we compare SWOP with alternative approaches based on real underlying phenomena, i.e. two synthetic data sets.

A. Estimation Results

To demonstrate the estimation result using SWOP, we run SWOP multiple times for every data set, and the estimation results with the highest compression rates were chosen for display.

The first data set has 372 measurements of salinity density off the coast of Southern California in the CalCOFI survey [29], based on which a 30×30 estimation map with $\tau = 0.2$ is generated. Figure 3(b) shows the estimation result based on the traditional centralized Kernel estimation while the result using SWOP with 0 order coefficients and the result based on the Voronoi diagram are shown in Fig.3(c) and Fig.3(a) respectively. In this example, the x-coordinate is the distance from the coast, y indicates the

depth of the sample, and a lighter point in Fig.3 indicates the saltier water.

The second set of estimation results based on a smaller data set from the Intel Lab is illustrated by Fig.4. In this data set, 48 point samples of the temperature were taken from a snapshot during an experiment in the Intel Lab[30]. A 30×30 map is estimated. The results based on the Voronoi diagram, the centralized Kernel estimation with $\tau = 11$ and SWOP with 0 order coefficients are shown by Fig.4(a), Fig.4(b) and Fig.4(c) respectively, where a darker point indicates the colder temperature.

Both real data sets only provide point samples, but the validation compared to the real underlying phenomena is not possible. Therefore, we use two synthetic data sets to test the effectiveness of SWOP. Two 401×401 continuous gray scale picture were synthetically generated as shown in Fig.5(a) and Fig.6(a). These two data sets can be interpreted as two different distributions of a "real phenomenon". For example, we can assume two gas leakages in the upper-left and lower-right corner of Fig.5(a). We set $\tau = 80$ to test the performance of SWOP based on 21×21 point samples taken from the underlying "phenomenon" at the interval of 20 pixels. Fig.5(b) and Fig.6(b) illustrate the results of centralized Kernel estimation. Fig.5(c) and Fig.6(c) show the SWOP estimation results with 0 order coefficients for the two synthetic data sets. For the fixed bandwidth, both centralized Kernel estimation and SWOP return a truthful estimation result on the synthetic data #1. For the second data set, the two small "gas leakages" are obscured, which indicates an over-smoothened result, and the result of SWOP based on the zero order Hermite coefficients is somewhat distorted compared with the centralized Kernel estimation result. However, we set the bandwidth fixed on purpose to compare the SWOP estimation results with other alternative estimation results based on their estimation qualities.

The estimation results based on Voronoi diagram show the layout and readings of sensor nodes directly, but the results are coarse compared with the results based on Kernel estimation. Furthermore, the cost of Voronoi diagram based approaches limits their application in constrained sensor networks. Whereas, even compared with the "real" phenomena, the results of SWOP still directly illustrate the real phenomenon distributions.

B. Error Evaluation

Spatial window queries access the distribution of underlying phenomena for a given region. An efficient in-network query processing targets to minimize the difference between results of the traditional centralized techniques and itself. The following tests are based on the average MSE from multiple runs. We first compared MSEs
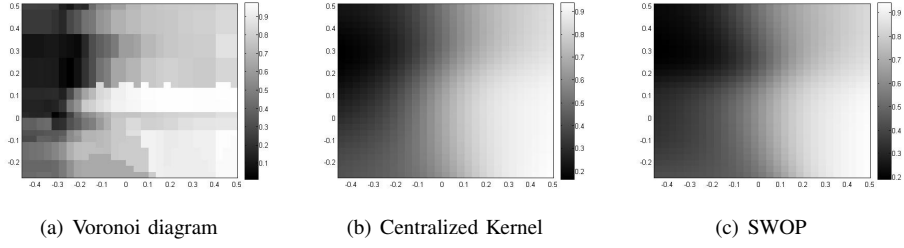
(a) Voronoi diagram   (b) Centralized Kernel   (c) SWOP

Fig. 3: Query results on the salinity data

figure

(a) Voronoi diagram   (b) Centralized Kernel   (c) SWOP
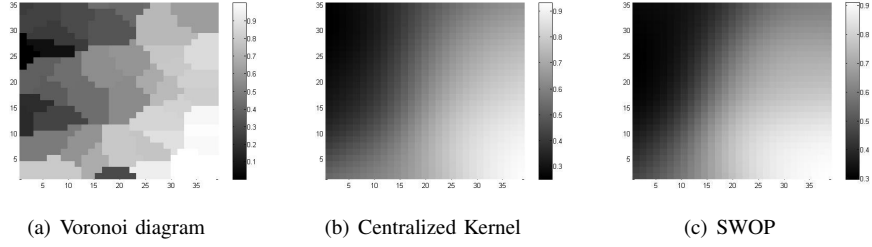
Fig. 4: Query results on the Intel lab data

figure

(a) Original Data   (b) Centralized Kernel   (c) SWOP

Fig. 5: Query results on the synthetic Data #1

figure

(a) Original Data   (b) Centralized Kernel   (c) SWOP
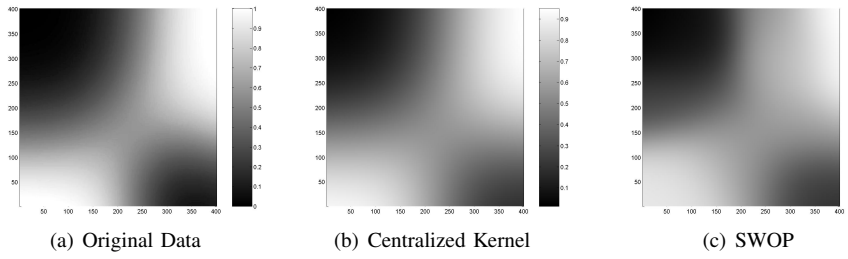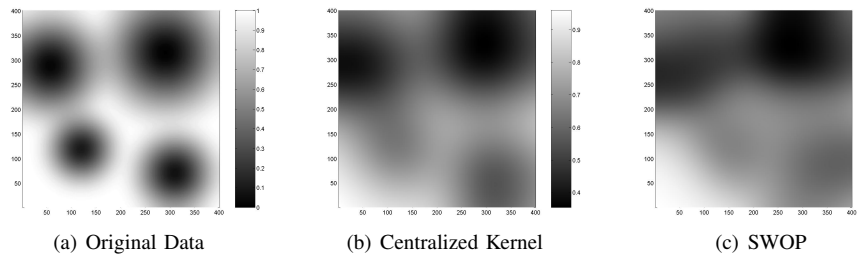
Fig. 6: Query results on the synthetic Data #2

figure

between the results of SWOP and centralized Kernel estimation as shown by Tab.III based on different truncating strategies. Tab.III confirms that by ordering the polynomial order of Hermite coefficients, SWOP achieves high quality results while relaxing the data requirement compared with taking the p^2 product by FGT. Aggregating more Hermite coefficients with higher polynomial-order decreases the difference between results of SWOP and the centralized

TABLE III: Cost and quality based on different truncating strategies

table

p	1	2	3	4	5	6	7	8	9	10
Number Of Total Terms In 2D Space based on different truncating strategies										
p-1 order	1	3	6	10	15	21	28	36	45	55
p terms	1	4	9	16	25	36	49	64	81	100
MSE on the salinity data based on different truncating strategies										
p-1 order	6.93E-04	5.20E-04	7.06E-05	2.01E-05	3.98E-06	8.67E-07	1.66E-07	2.78E-08	5.22E-09	7.18E-10
p terms	6.93E-04	4.53E-04	2.48E-05	1.06E-05	7.88E-07	2.54E-07	1.86E-08	4.18E-09	2.83E-10	4.55E-11
MSE on the Intel Lab data based on different truncating strategies										
p-1 order	1.03E-03	3.19E-04	1.01E-04	1.39E-05	4.76E-06	5.30E-07	1.27E-07	1.55E-08	2.18E-09	3.01E-10
p terms	1.03E-04	3.19E-04	4.55E-05	5.46E-06	1.18E-06	1.21E-07	1.62E-08	1.79E-09	1.33E-10	1.76E-11
MSE on the synthetic data #1 based on different truncating strategies										
p-1 order	7.51E-04	5.24E-04	5.39E-05	9.08E-06	2.19E-06	3.42E-07	5.27E-08	6.47E-09	8.35E-10	7.54E-11
p terms	7.51E-04	4.96E-04	1.01E-04	1.67E-05	5.96E-06	8.55E-07	2.33E-07	2.74E-08	6.36E-09	6.17E-10
MSE on the synthetic data #2 based on different truncating strategies										
p-1 order	2.58E-03	3.64E-04	3.03E-04	2.42E-05	1.29E-05	1.13E-06	4.12E-07	3.70E-08	1.04E-08	8.76E-10
p terms	2.58E-03	3.98E-04	1.78E-04	1.27E-05	4.72E-06	3.47E-07	8.44E-08	6.19E-09	1.23E-09	7.32E-11

Kernel estimation. Since the maximal MSE between results of SWOP and the centralized Kernel estimation results based on zero-order Hermite coefficients are around 10^{-3}, we performed other quality tests based on the zero-order Hermite coefficients.

TABLE IV: Mean squared errors relative to "real" values

table

Data set	Kernel	SWOP
Synthetic #1	4.6E-03	7.16E-03
Synthetic #2	3.23E-02	4.40E-02

While the first two real data sets only provide us point samples of a realistic underlying phenomenon, the two synthetic data sets give us a chance to compare the estimated results with "real" values as shown by Tab.IV. The method introduced by [32] can help users to find the optimal bandwidth according to different phenomenon distributions. Based on our choice for the first synthetic data set, the mean squared errors between SWOP result and the "real" phenomenon are around 10^{-3}. The SWOP result of #1 set is reliable for many practical purposes. We fixed the bandwidth for the second synthetic data to test SWOP against alternative approaches, although the MSE on the second synthetic data indicates an over-smoothened result.

C. Cost Evaluation

TABLE V: Required data size for each cluster(in bit)

table

Data set	# of clusters	Raw data	SWOP
Salinity	21	3475.39	253.85
Intel-lab	8.8	1093.12	249.7
Synthetic #1	23.1	3572.66	251.4
Synthetic #2	22.7	3730.04	252.3

In our tests, we use one double (64*bits*) to present a sensor reading and two doubles (128*bits*) to present a

sensor node identity, i.e. its location. We recorded the average number of clusters and the average size of raw data and SWOP data for each cluster based on zero order Hermite coefficients from multiple independent tests on each data set, as shown by Tab.V. The clustering algorithm plays an important role in SWOP for the compression gain. After being clustered, a non-head node requires 192 bits to send its reading and ID to the cluster head. The message size for each cluster head to present its cluster members depends on the chosen order of Hermite coefficient. For the zero-order, each cluster head needs 256 bits to represent its member nodes for both the numerator and denominator in Eq.9. Since small clusters just send their raw readings, the average message size is a little smaller than 256 bits as show by Tab.V. Compared with transmitting raw data for each cluster, SWOP saves 94% messages. The total communication cost of a network depends on different communication protocols and network layouts, and is hard to be simulated. Users can find relevant evaluation results about it from [26], [28].

D. Comparison With Alternative Approaches

1) Wavelet And Delta Compression: Any compression algorithm can be applied in clustering protocols to compress raw sensor readings for each cluster. We implemented Haar wavelets and allow cluster heads to transform raw readings and node IDs into wavelets. Tab.VI illustrates the experiment results on the two real data sets for different wavelet coefficient settings. An advantage of wavelets is that they can present data in different scales and compress lossless data based on which we can apply any analytical models. As shown in Tab.VI, Haar wavelets can compress lossless data in about 60% size of the raw data for each cluster. In our experiments, we only compared the centralized Kernel estimation results based on wavelet data with the Kernel estimation results on

TABLE VI: Evaluation on wavelets

table

Coefficient threshold		Data size	MSE
readings	node ID		
Intel-Lab data			
0	0	698.98	0
0.2	0	449.71	9.53E-04
0.4	0	391.79	1.71E-02
0	6	661.98	1.23E-03
0	10	649.04	2.63E-03
0.3	6	310.82	8.53E-03
0.4	6	317.28	2.49E-02
0.3	8	323.61	8.6E-03
0.4	8	329.81	2.92E-02
Salinity data			
0	0	2896.26	0
0.2	0	1967.57	9.63E-04
0.4	0	1919.22	1.11E-02
0	0.10	1529.13	1.93E-03
0	0.2	1534.45	7.43E-03
0.2	0.1	651.84	1.97E-03
0.3	0.1	687.52	7.43E-03
0.2	0.15	619.79	6.43E-03
0.3	0.15	594.22	1.33E-02

original data. By eliminating small wavelet coefficients, we can achieve higher compression rates, but degrade the estimated results. However, to achieve a similar quality of SWOP, wavelet-based methods require a larger data size than that SWOP does. More tests on the synthetic data sets and Delta-compression show very similar results of wavelets, therefore we exclude the detailed comparison about them due to the space limitation. By evaluating wavelets, Delta-compression and SWOP, we can conclude that SWOP requires a smaller size of data but still returns high quality estimation results.

2) Spatial Regression: Since we fixed the bandwidth for both synthetic data sets, we can compare SWOP with different 2D spatial regression methods on the synthetic data sets based on different estimation qualities. We did our tests to evaluate the estimation results against the "real" phenomena and the cost of processing alternative approaches in the network.

TABLE VII: Evaluation on 2D polynomial regression

table

Polynomial Order	MSE	# of $f()s$
Synthetic data #1		
1	7.5E-02	3
2	1.2E-02	6
3	4.8E-03	10
4	1.5E-03	15
Synthetic data #2		
1	6.2E-02	3
2	6.0E-02	6
3	4.9E-02	10
4	2.6E-02	15

We run different 2D spatial regression methods in a traditional centralized setting on all raw data. The results based on different orders of polynomial regressions is shown by Tab.VII. With higher orders of polynomial equa-

tions, the estimated results get better. To achieve a similar quality of SWOP with the current bandwidth setting, a 2D spatial polynomial regression requires 10 or more basis functions for both synthetic data sets. For Kernel regression, we test different numbers of kernels based on different kernel functions separated at fixed intervals with different bandwidths. Tab.VIII illustrates the minimal MSE based on different numbers of kernels and different kernel functions. Tab.VIII also shows the chosen bandwidth and kernel-center interval for the different kernel functions to return the best estimation results based on different numbers of kernels. To achieve a similar quality of SWOP, the Kernel regression requires 9 or more kernels. Fig.7 and Fig.8 show the estimation results based on the cubic polynomial, and the best estimation results based on 9 cone kernels and 9 Gaussian kernels for synthetic data #1 and #2 respectively.

Generally, both regression estimation methods require 9 or more basis functions to achieve a similar or better quality of SWOP. To return the final estimation results, we need at least $(81+9) \cdot k_i$ data from the network. Applying several types of kernel functions deceases the size of data exchanged among neighboring nodes, but the estimation results are not very smooth due to the discontinuity of kernel functions, i.e. the estimation results based on cone kernels Fig.7(b). On average, for both synthetic data, SWOP returns around 23 clusters, and requires a similar size of data, about $23 \cdot 4 \cdot k_i$, from the network for a similar quality compared with the 2D spatial regression methods. However, almost all nodes involved in regression methods need to receive and send the same large size of data. In SWOP, only the cluster heads near to the central base or a micro-server need to communicate with the large-sized messages. The nodes within a cluster and the nodes at the bottom on a routing tree in SWOP relax their communication costs. Furthermore, for the current cluster radius setting $0.9(80) = 72$ and the current spatial window size 401×401, a compact clustering pattern should contain less than 9 clusters; whereas the distributed clustering algorithms do not return an ideal clustering pattern. SWOP can achieve a higher compression gain by applying more sophisticated clustering methods.

Regression estimation methods focus on minimizing global errors, while SWOP and non-parametric estimation methods focus on revealing local variations. If we compare the estimation results of SWOP and regression estimation methods to the "real" underlying phenomena, the local change is better preserved by SWOP than by regression estimation methods for the similar global quality, MSE. For example, in Fig.8(a), one of the small peaks has totally disappeared.

TABLE VIII: Evaluation on Kernel regression

table

# of kernels	Block kernel			Cone kernel			Gaussian kernel		
	Min MSE	Kernel Interval	Bandwidth	Min MSE	Kernel Interval	Bandwidth	Min MSE	Kernel Interval	Bandwidth
Synthetic data #1									
16	8.37E-03	110	165	4.06E-04	130	195	4.60E-04	110	95
9	2.09E-02	140	210	1.72E-03	140	210	1.15E-03	160	130
4	6.84E-02	200	300	1.37E-02	210	315	1.19E-02	210	265
Synthetic data #2									
16	2.96E-02	120	260	2.83E-02	110	325	3.85E-02	130	65
9	3.78E-02	170	255	4.13E-02	160	310	4.19E-02	150	75
4	6.26E-02	200	300	5.82E-02	200	300	5.89E-02	170	165

(a) Polynomial   (b) Cone Kernel   (c) Gaussian Kernel

Fig. 7: Alternative estimations on the synthetic data #1

figure

(a) Polynomial   (b) Cone Kernel   (c) Gaussian Kernel
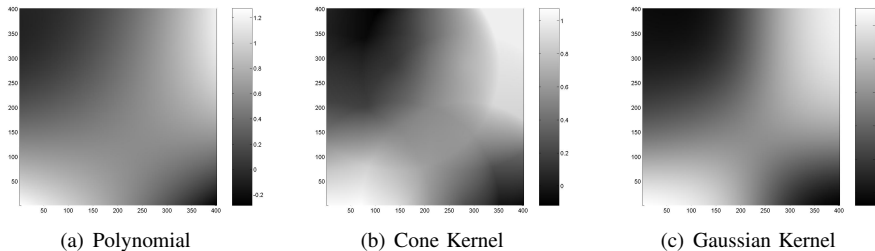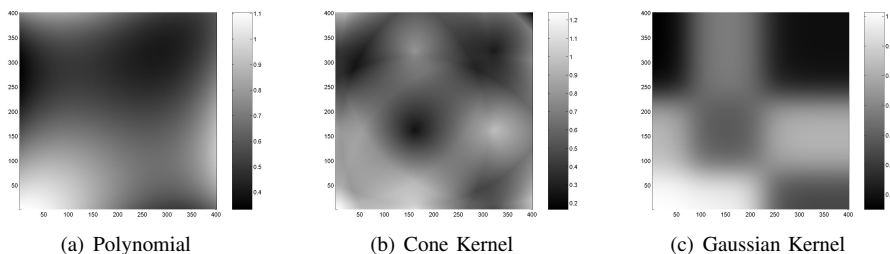
Fig. 8: Alternative estimations on the synthetic data #2

figure

VII. Related Work

As a basis for many protocols and a key element for sensor data processing, sensor node clustering algorithms are important in the field of sensor networks. LEACH[24] and HEED[23] are two of the most efficient algorithms available. LEACH first introduced a probability selection to choose the cluster head. Each node in LEACH can be the cluster head, and the probability of being a cluster head is predefined. Nodes rotate the roles of being a non-head member or a cluster head to save energy. HEED [23] improves the selection procedure by using other metrics, such as the remaining energy, to determine a dynamic probability. HEED has several additional advantages over LEACH, such as supporting multi-hop clustering.

Due to the constrained environment of sensor networks, sensor DBMSs favor aggregation queries. TAG[1] provides such a framework for this type of queries. TAG uses a tree structure to connect sensor nodes. Each node only processes and aggregate the partial results from its descendants, so the communication cost can be minimized and the messages are often kept constant. Trigoni et al. provide an optimization strategy for multiple aggregation queries in sensor networks [26], [28]. They categorized different types of aggregation queries and provided different optimization strategies to reform the communication topology.

As a result of the built-in spatial properties, sensor readings naturally support spatial queries. Simple aggregation queries[1], [2], however, often fail to generate correct answers for spatial aggregation queries if the nodes are not evenly distributed. Current approaches applied basic spatial interpolation methods, e.g. Voronoi diagram and TIN[10], to estimate the aggregation results. Kernel regression method has also been utilized to support spatio-temporal

queries in sensor networks[14]. To find the weights for predefined kernel functions in the network, however, a sensor network requires iterations of communication [14], which deteriorates the network performance. In [13], a 2D polynomial regression was used to control the contour map quality.

In-network compression techniques are also important to sensor networks to reduce the in-network resource consumption. In [33], wavelet based compression has been introduced, where a sensor network can do pairwise comparison between sensor readings to generate discrete wavelets and reduce in-network transmitted data.

VIII. Conclusion And Future Work

In this paper, we presented a novel in-network estimation technique of spatial window queries, SWOP, designed for answering direct queries on continuous phenomena based on Kernel estimation method. SWOP breaks the entangled links between estimation points and sensor nodes by utilizing Hermite expansion. In SWOP, a small number of Hermite coefficients represents all information about the total invoked sensor nodes including their locations and readings. Hence, SWOP minimizes the size of data and relaxes the computation complexity, compared with a centralized solution and an ordinary distributed solution. Our simulation of SWOP tested the data sets of real phenomena, and synthetic data. The result of the simulation shows that SWOP competes well with other approaches by relaxing the resource consumptions and still providing high quality query results.

We will combine the temporal and spatial aspects together for the future version of SWOP, which is not covered in this paper.

References

[1] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," *ACM SIGOPS*, vol. 36, no. SI, pp. 131–146, 2002.

[2] J. M. Hellerstein, W. Hong, S. Madden, and K. Stanek, "Beyond average: Toward sophisticated sensing with queries.," in *IPSN*, pp. 63–79, 2003.

[3] A. Demers, J. Gehrke, R. Rajaraman, N. Trigoni, and Y. Yao, "The cougar project: a work-in-progress report," *SIGMOD Record*, vol. 32, no. 4, pp. 53–59, 2003.

[4] P. Bonnet, J. Gehrke, and P. Seshadri, "Querying the physical world," *IEEE Personal Communication*, vol. 7, pp. 10–15, 2000.

[5] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *ACM SenSys*, pp. 51–63, 2005.

[6] J. Burrell, T. Brooke, and R. Beckwith, "Vineyard computing: Sensor networks in agricultural production," *IEEE Pervasive Computing*, vol. 3, pp. 38–45, Jan. 2004.

[7] M. Duckham, S. Nittel, and M. Worboys, "Monitoring dynamic spatial fields using responsive geosensor networks," in *ACM GIS*, pp. 51–60, 2005.

[8] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos, "Compressing historical information in sensor networks," in *ACM SIGMOD*, pp. 527–538, 2004.

[9] A. Jain, E. Y. Chang, and Y.-F. Wang, "Adaptive stream resource management using kalman filters," in *ACM SIGMOD*, pp. 11–22, 2004.

[10] M. Sharifzadeh and C. Shahabi, "Supporting spatial aggregation in sensor network databases," in *ACM GIS*, pp. 166–175, 2004.

[11] F. Aurenhammer and Klein, *Handbook of Computational Geometry*, ch. 5. Elsevier Science Pub Co, 1st ed., January 1, 2000.

[12] B. Harrington and Y. Huang, "In-network surface simplification for sensor fields," in *ACM GIS*, pp. 41–50, 2005.

[13] W. Xue, Q. Luo, L. Chen, and Y. Liu, "Contour map matching for event detection in sensor networks," in *ACM SIGMOD*, pp. 145–156, 2006.

[14] C. Guestrin, P. Bodík, R. Thibaux, M. A. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data.," in *IPSN*, pp. 1–10, 2004.

[15] V. Delouille, R. Neelamani, and R. G. Baraniuk, "Robust distributed estimation in sensor networks using the embedded polygons algorithm.," in *IPSN*, pp. 405–413, 2004.

[16] J. Racine, "Parallel distributed kernel estimation," *Computational Statistics and Data Analysis*, vol. 40, no. 2, pp. 293–302, 2002.

[17] M. Broadie and Y. Yamamoto, "Application of the fast gauss transform to option pricing," *Management Science*, vol. 49, no. 8, pp. 1071–1088, 2003.

[18] C. Yang, R. Duraiswami, N. A. Gumerov, and L. Davis, "Improved fast gauss transform and efficient kernel density estimation," in *ICCV*, pp. 464–471, IEEE Computer Society, 2003.

[19] L. Greengard and J. Strain, "The fast gauss transform," *SIAM Journal on Scientific and Statistical Computing*, vol. 12, no. 1, pp. 79–94, 1991.

[20] B. J. C. Baxter and G. Roussos, "A new error estimate of the fast gauss transform," *SIAM Journal on Scientific Computing*, vol. 24, no. 1, pp. 257–259, 2002.

[21] M. Bern and D. Eppstein, "Approximation algorithms for geometric problems," pp. 296–345, 1997.

[22] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 1 ed., 2001.

[23] O. Younis and M.-S. Fahmy, "Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, 2004.

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

[25] G. Jin and S. Nittel, "Udc: A self-adaptive uneven clustering protocol for dynamic sensor networks," in *International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, 2005.

[26] N. Trigoni, Y. Yao, A. J. Demers, J. Gehrke, and R. Rajaraman, "Multi-query optimization for sensor networks.," in *DCOSS* (V. K. Prasanna, S. S. Iyengar, P. G. Spirakis, and M. Welsh, eds.), vol. 3560 of *Lecture Notes in Computer Science*, pp. 307–321, Springer, 2005.

[27] R. Nowak and U. Mitra, "Boundary estimation in sensor networks: Theory and methods.," in *IPSN*, pp. 80–95, 2003.

[28] N. Trigoni, Y. Yao, A. J. Demers, J. Gehrke, and R. Rajaraman, "Multi-query optimization for sensor networks," In TR2005-1989, Cornell University, 2005.

[29] A. Bucklina, P. H. Wiebeb, S. B. Smolenacka, N. J. Copleyc, and M. E. Clarke, "Integrated biochemical, molecular genetic,and bioacoustical analysis of mesoscale variability of the euphausiid nematoscelis difficilis in the california current," *Deep-Sea Research*, vol. 49, pp. 437–462, 2002.

[30] http://berkeley.intel research.net/labdata/.

[31] O. V. Lepskii and V. Spokoiny, "Optimal pointwise adaptive methods in nonparametric estimation," tech. rep., undated.

[32] V. C. Raykar and R. Duraiswami, "Fast optimal bandwidth selection for kernel density estimation," in *Proceedings of the sixth SIAM*

International Conference on Data Mining (J. Ghosh, D. Lambert, D. Skillicorn, and J. Srivastava, eds.), pp. 524–528, 2006.

[33] J. Hellerstein and W.Wang, "Optimization of in-network reduction," in *VLDB-Workshop DMSN*, pp. 166–175, 2004.