Information Sciences 243 (2013) 20–38

Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A new kernelized approach to wireless sensor network localization

Jaehun Lee, Wooyong Chung, Euntai Kim *

School of Electrical and Electronic Engineering, Yonsei University, 134 Sinchon-dong, Seodaemun-gu, Seoul 120-749, Republic of Korea

ARTICLE INFO

Article history:
Received 26 July 2010
Received in revised form 1 May 2012
Accepted 19 April 2013
Available online 25 April 2013

Keywords:
Wireless sensor network
Range-free localization
Regression
Kernel

ABSTRACT

In this paper, a new approach to range-free localization in Wireless Sensor Networks (WSNs) is proposed using nonlinear mapping, and the kernel function is introduced. The localization problem in the WSN is formulated as a kernelized regression problem, which is solved by support vector regression (SVR) and multi-dimensional support vector regression (MSVR). The proposed methods are simple and efficient in that no additional hardware is required for the measurements, and only proximity information and position information of the anchor nodes are used for the localization. The proposed methods are composed of three steps: the measurement step, kernelized regression step, and localization step. In the measurement step, the proximity information of the given network is measured. In the regression step, the relationships among the geographical distances and the proximity among sensor nodes is built using kernelized regression. In the localization step, each sensor node finds its own position in a distributed manner using a kernelized regressor. The simulation result demonstrates that the proposed methods exhibit excellent and robust location estimation performance.

1. Introduction

Wireless Sensor Networks (WSNs) have been proposed for various applications including object tracking [24,14], disaster management [23], and smart home applications [12]. Typically, WSNs are composed of a large number of *sensor nodes* and a relatively small number of *anchor nodes*. Sensor nodes are very small, and their locations are unknown in advance. By contrast, the anchor nodes have additional hardware, and their locations are known a priori. In many applications, localization of the sensor nodes in a given network is of crucial importance. The simplest possible solution to localization would be to attach a global positioning system (GPS) [6] to all of the sensor nodes, but this would be very costly or even impossible, since the sensor nodes have limited resources. Since the position information of all sensor nodes represents one of the most essential pieces of information in many WSN applications [4,9], it is important to estimate the geographic locations of all sensor nodes in the given WSN using only the minimal location information of the anchor nodes and some measurements among the nodes.

Many studies have been conducted on the localization problem in WSNs [2,5,8,11,15–17,19,21,26,28,30,35–37], and they can be divided into two types: range-based algorithms and range-free algorithms. The range-based localization algorithms utilize range measurements such as distance measurements and angle measurements from the anchor nodes. The distance can be measured by a received signal strength indicator (RSSI), time of arrival (TOA), or time difference of arrival (TDOA), and the angle can be measured by the angle of arrival (AOA) [17]. The range-based localization algorithms yield relatively precise

* Corresponding author.
E-mail address: etkim@yonsei.ac.kr (E. Kim).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.04.024

estimations of location, but require additional hardware, and their costs are relatively high [26]. By contrast, range-free localization algorithms use only the connectivity information among sensors, i.e., "who is within the communications range of whom" [28]. Thus, the range-free approaches do not require any additional hardware for measurements. In the case of large-scale WSN applications involving hundreds or thousands of sensors, the range-free algorithm would be preferred, since it is impractical to attach additional hardware to all of the sensors. However, one of the most fatal drawbacks of the range-free algorithms is that the accuracy of the location estimation is somewhat poor because of the absence of any distance measurements between the nodes. The motivation of this paper is to develop a range-free localization algorithm with improved location estimation accuracy. To address this problem, the machine learning technique is applied to the localization.

In this paper, two range-free localization algorithms, called Localization through Support Vector Regression (LSVR) and Localization through Multi-dimensional Support Vector Regression (LMSVR), are proposed. The proposed methods are composed of three steps: the measurement step, kernelized regression step, and localization step. First, the proximity information of the given network is measured. Second, the relationships between geographical distances or locations and proximity among sensor nodes are modeled using a kernelized regression approach. Finally, each sensor node finds its own position in a distributed manner.

The rest of this paper is organized as follows: In Section 2, related work about range-free localization algorithms in WSNs is described. In Section 3, the motivation of this paper is explained and two new localization methods are proposed. In Section 4, various simulations are conducted and the results of the proposed methods are compared with those of previous methods. Finally, conclusions are drawn in Section 5.

2. Background

Many studies have been conducted on range-free localization in WSN. In a pioneering work, Niculescu and Nath developed the DV-hop approach, in which the average hop distance was computed and applied to the sensor nodes for localization [21]. Bulusu et al. proposed a range-free, proximity-based, coarse-grained localization algorithm called the centroid localization algorithm (CLA) [2]. In this method, the anchor nodes broadcast their positions, and each sensor node computes its position with respect to the connected anchor nodes. He et al. proposed the APIT localization algorithm [5]. APIT estimates locations by dividing the environment into triangular regions between anchor nodes, while narrowing down the likely area according to the presence of each sensor node inside or outside of those triangles. Liu et al. proposed an improved centroid localization algorithm (ICLA) based on APIT and the quality of a perpendicular bisector [16]. In ICLA, sensor nodes are classified into one of four groups, a normal node, side node, hypoisolated node, and isolated node, and each node can be localized with different rules based on the classification result. Lim et al. developed a proximity-distance map (PDM) algorithm to handle anisotropic networks [15]. In this work, the authors regarded localization as an embedding problem that linearly maps the geographic distance into the proximity measurement, and proceeded to solve this problem by defining the linear mapping matrix. More specifically, PDM derives the linear transformation matrix *T* that enables mapping from the proximity vector to the distance vector. For outdoor open-air environments Zhong et al. proposed the regulated signature distance (RSD), which can be easily embedded in connectivity-based localization algorithms to improve the localization accuracy [37].

On the other hand, several kernel-based methods for location estimation in WSNs have been reported, using either classification or regression strategies. Pan et al. proposed the localization algorithm based on Kernel Canonical Correlation Analysis [22], and Brunato et al. applied the SVM to the localization problem [1]. Tran et al. reported SVM-based range-free WSN localization in [32], which will be explained in a subsequent section. Moreover, Kuh et al. proposed the kernel regression algorithm, which uses the recursive least squares algorithm to solve the problem [10]. Honeine et al. proposed the matrix regression method using a kernel approach [7]. Interestingly, Shilton et al. applied several existing support vector regression (SVR) algorithms using received signal strength (RSS) as measurements [30]. In the paper, however, it was assumed that every sensor should be connected to almost all sensor and anchor nodes in the given network for successful SVR modeling, but the assumption is impractical to many real-world applications (Lee et al. proposed the localization based on multi-dimensional SVR in [13]).

These kernel-based algorithms provide quite accurate localization results. All of these methods except [32], however, use distance measurements, including RSS measurements between the sensors and anchors, to solve the localization problem. Motivated by these kernel-based methods, new kernelized approaches without the use of any distance measurements are proposed in this paper.

2.1. WSN localization by kernelized classification

One interesting range-free WSN localization algorithm was reported by Tran and Nguyen [32], called LSVM (Localization based on the Support Vector Machine). The basic idea of LSVM is to transform the WSN connection by a higher-order nonlinear transform and view the localization problem as a classification problem in the transformed domain. That is, the space of interest is decomposed into subspaces, and each sensor node is assigned to subspaces in the transformed domain by the classification approach shown in Fig. 1.

More specifically, suppose that there are *N* sensor nodes in a 2D geographic area $[0,D] \times [0,D]$, and *k* of them are beacon nodes ($k \ll N$), the location of which are known *a priori*. LSVM decomposes the given space $[0,D] \times [0,D]$ into

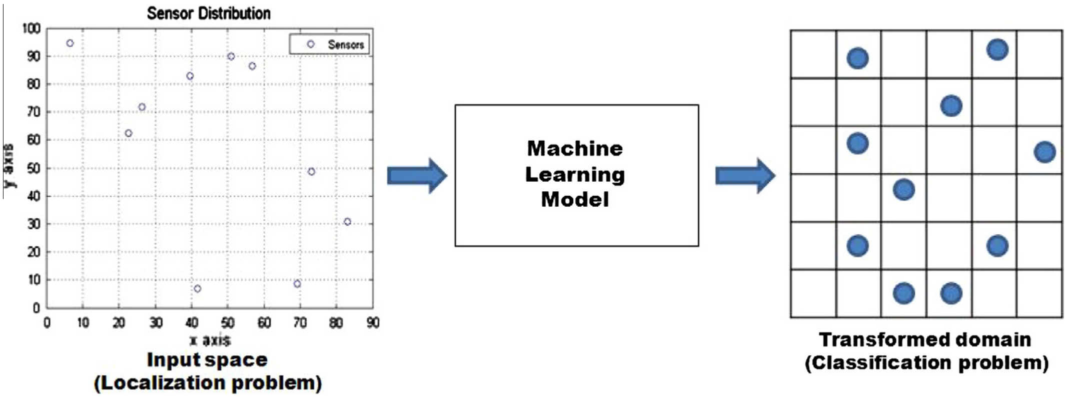


Fig. 1. Transforming the localization problem into a kernelized classification problem.

$M^2 = 2^{2m}$ square subspaces with the size $[0, \frac{D}{2^m}] \times [0, \frac{D}{2^m}]$, and assigns one subspace for each sensor node using the Support Vector Machine (SVM) model. The beacon nodes are then used as the training set of the SVM, since their positions are known *a priori*. Each sensor node is assigned to one subspace as a result of the SVM. Fig. 2 presents an example of the SVM decision tree for an x-dimensional subspace.

The final location of each sensor node is estimated as the center of the corresponding subspace. There are several other studies about localization in WSN using SVM [20,34]. However, these localization algorithms based on SVM only estimate the approximate regions where the sensor node should be placed, and are not able to identify the precise (x,y) coordinates of the node.

3. WSN localization by kernelized regression

3.1. Motivation

Most of the previous range-free methods do not exploit the global or topological information of the whole sensor network, and they only use the local neighborhood information around a sensor node. The use of only local information around a sensor node may degrade the localization performance when WSN is anisotropic. Interestingly, PDM utilizes the global information of the entire WSN by formulating the localization into a regression problem [15]. In PDM, however, fitting the relationship between the proximity and geometry in the WSN causes substantial error in the localization. In this paper, by introducing the kernel technique and exploiting the global information of the WSN, the proposed methods develop a non-linear model to represent the relationship between the proximity and geometry of the WSN.

The proposed methods are also different from those in [32] in that the WSN localization is viewed not as a kernelized classification problem, but as a kernelized regression problem. While [32] tackles the WSN localization problem by considering it as a two-dimensional subspace classification problem, the proposed methods solve the same problem by viewing it as a regression problem. In contrast to other kernel-based methods for location estimation in WSN using RSSI measurements, the proposed methods only consider the connectivity among the nodes. Two methods are proposed in this paper.

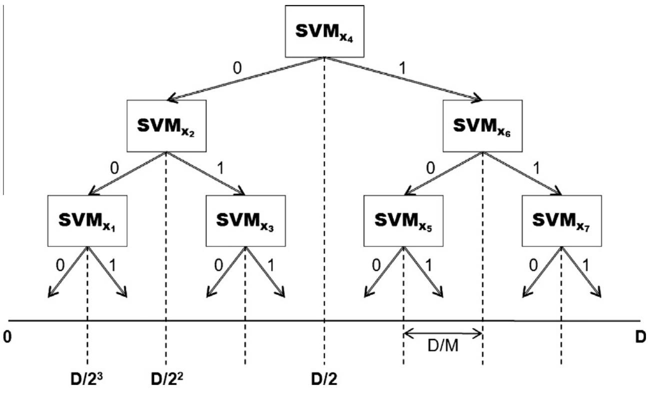


Fig. 2. An example of an SVM decision tree for an x-dimensional subspace ($m = 3$).

(a) LSVR: One-dimensional distance regression using the connectivity measurements and estimation of the sensor node locations via multilateration.
(b) LMSVR: Two-dimensional location regression using the connectivity measurements.

Fig. 3 shows the entire process of the proposed kernelized approaches to the WSN localization problem.

The motivation of this paper can be further considered from the perspective of machine learning theory. Most SVRs are one-dimensional [18], and their extension to a higher dimension is not straightforward. In this paper, two-dimensional SVR taken from [25] is employed to localize the sensor nodes without using multilateration. The lack of a multi-dimensional SVR comes from the fact that direct formulation of the SVR in a high dimension results in the non-affine constraint in the primal space, and thus it cannot be formulated as a convex optimization in a dual space. The two-dimensional SVR used herein circumvents these difficulties by transforming the optimization problem into a linear equation using a representation theorem. In the following subsections, the localization problem in the WSN is formulated into a regression problem, and the details of the proposed solutions are presented.

3.2. Problem formulation into regression

Let us consider a sensor network $S = \{S_1, S_2, \ldots, S_{M+N}\}$ with M anchor nodes and N sensor nodes ($M << N$), and denote the position of each node as

$$\mathbf{pos}(S_i) = (x_i, y_i)^T \quad \text{for } i = 1, \ldots, M + N. \tag{1}$$

Here, the space is assumed to be two-dimensional for simplicity, but the proposed methods can also be applied to a three-dimensional space. The positions of M anchor nodes $S_i \in \mathrm{A}$ are known, but the positions of the other N sensor nodes $S_j \in \Sigma$ are unknown, where $\mathrm{A} \triangleq \{S_i | i = 1,2, \ldots, M\}$ and $\Sigma \triangleq \{S_j | j = M + 1, M + 2, \ldots, M + N\}$. It is assumed that the only available measurement is the proximity information denoting the number of hops between all of the nodes. The geographic distance between two nodes S_i and S_j is defined as

$$d(S_i, S_j) = \|\mathbf{pos}(S_i) - \mathbf{pos}(S_j)\| = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \in \Re, \tag{2}$$

and the proximity information $p(S_i, S_j) \in \mathbb{Z} \triangleq \{0, 1, 2, \ldots\}$ between S_i and S_j implies the number of hop counts between the two nodes. The localization problem can then be formulated as

$$\begin{aligned} &\text{Estimate } \mathbf{pos}(S_\alpha) \\ &\text{given } \mathbf{pos}(S_i), d(S_i, S_j), \text{ and } p(S_k, S_l) \end{aligned} \tag{3}$$

where S_i, $S_j \in A$, S_k, $S_l \in A \cup \Sigma$, and $S_\alpha \in \Sigma$. The basic idea of this approach is to reformulate the localization problem into a kernelized regression problem in which the location of the sensor nodes $S_\alpha \in \Sigma$ or the related quantity is predicted from the proximity information $p(S_i,S_j)$ by

$$f(\mathbf{p}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{p}) + b, \tag{4}$$

where $\mathbf{p}$ is the collection of the proximity information $p(S_i,S_j)$, and $\boldsymbol{\varphi}(\cdot)$ is a nonlinear mapping acting on the proximity vector. The details of Eq. (4) are presented in the following subsection. In this paper, two methods are proposed for the localization: (a) localization through support vector regression (LSVR) and (b) localization through multi-dimensional support vector regression (LMSVR). The details of the proposed methods are presented in the next subsections.

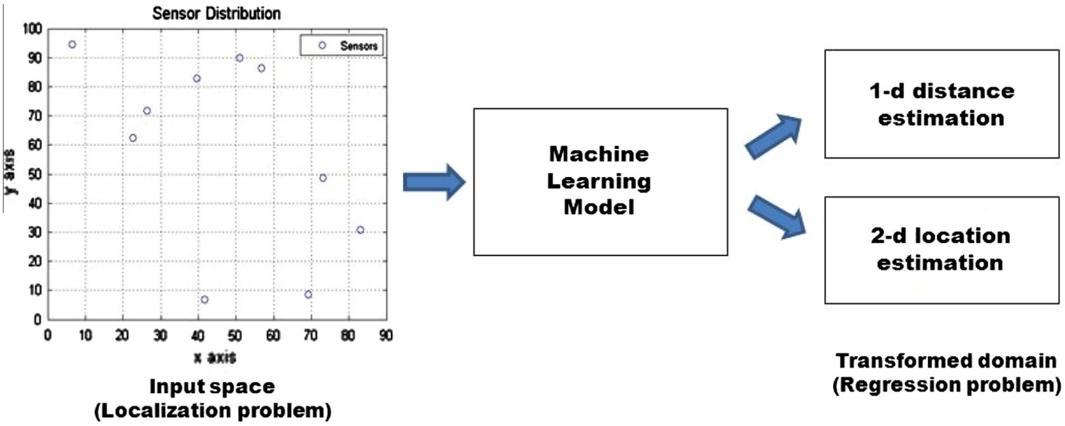


Fig. 3. Transforming the localization problem into a kernelized regression problem.

3.3. Localization through Support Vector Regression (LSVR)

The LSVR is composed of the following three steps:

STEP 1 (Measurement Step): Each anchor node $S_i \in$ A and sensor node $S_j \in \Sigma$ exchanges the hop count information with one another, and each anchor node S_i transmits its position information to the other anchor nodes.
STEP 2 (SVR Step): Each anchor node $S_i \in$ A estimates its own distance model SVR_i using SVR, and broadcasts it to all sensor nodes $S_j \in \Sigma$. Consequently, M distance models ($SVR_1, SVR_2, \ldots, SVR_M$) are assigned to all sensor nodes S_j.
STEP 3 (Localization Step): Each sensor node $S_j \in \Sigma$ computes the distances from all anchor nodes based on the distance models ($SVR_1, SVR_2, \ldots, SVR_M$) given in the previous SVR step, and finally estimates its own position by the multilateration method in a distributed manner.

The communication costs of the entire proposed LSVR are $O(MN)$ messages. The details of above three steps are discussed in the following subsections.

3.3.1. Measurement step

Let

$$\mathbf{p}_i = \begin{bmatrix} p_{i,1} & \cdots & p_{i,M} \end{bmatrix}^T \in \mathbb{Z}^M \tag{5}$$

be the proximity vector between the anchor node $S_i \in$ A and the other anchors, where $p_{i,j} = p(S_i, S_j)$, S_i, $S_j \in A$, and $p_{i,i} = 0$; Z is the set of integers. The overall proximity information between anchor nodes can then be represented by

$$\mathbf{P} = [\,\mathbf{p}_1 \quad \cdots \quad \mathbf{p}_M\,] \in \mathbb{Z}^{M\times M}. \tag{6}$$

Similarly, let

$$\mathbf{d}_i = \begin{bmatrix} d_{i,1} & \cdots & d_{i,M} \end{bmatrix}^T \in \Re^M \tag{7}$$

be the geographic distance vector between the anchor node $S_i \in$ A and the other anchors, where $d_{i,j} = d(S_i, S_j)$, S_i, $S_j \in A$, and $d_{i,i} = 0$. The overall distance information between anchor nodes is summarized as

$$\mathbf{D} = [\,\mathbf{d}_1 \quad \cdots \quad \mathbf{d}_M\,] \in \Re^{M\times M}. \tag{8}$$

Let $\bar{\mathbf{p}}_k$ be the proximity vector between the kth sensor node $S_k \in \Sigma$ and all anchor nodes $S_i \in A$. Then,

$$\bar{\mathbf{p}}_k = [\,\bar{p}_{k,1} \quad \cdots \quad \bar{p}_{k,M}\,]^T \in \mathbb{Z}^M, \tag{9}$$

where $\bar{p}_{k,i} = p(S_k, S_i)$, $S_k \in \Sigma$, and $S_i \in A$.

The actual communication procedure for the measurement step is almost the same as that in [32]. First, each anchor node $S_i \in A$ broadcasts a HELLO message to all of the other nodes, including both sensor and anchor nodes $S_l \in S$ such that each anchor node $S_i \in A$ and sensor node $S_k \in \Sigma$ knows its proximity vector $\mathbf{p}_i$ and $\bar{\mathbf{p}}_k$, respectively. Each anchor node S_i then sends an INFO message including its proximity vector $\mathbf{p}_i$ and location information $\mathbf{pos}(S_i)$ to the other anchor nodes $S_j \in A$, such that each anchor node builds **P** and **D**. The communication costs of the measurement step are $M(M + N - 1)$ HELLO messages and $M(M - 1)$ INFO messages. Since $M \ll N$ in general, the total communication cost is $O(MN)$ in the measurement step.

3.3.2. Training step (SVR Step)

In the training step, each anchor node $S_i \in A$ builds a kernelized regression model to predict the distance to any of the other nodes. That is, each anchor node S_i collects M training data pairs, i.e., $(\mathbf{p}_j, d_{i,j}) \in \mathbb{Z}^M \times \Re$ $(j = 1, 2, \ldots, M)$, which relates anchor S_i to all anchor nodes $S_j \in A$, including itself. The anchor node S_i then trains a nonlinear function

$$f_i(\mathbf{p}_j) = \mathbf{w}_i^T \boldsymbol{\varphi}(\mathbf{p}_j) + b_i, \tag{10}$$

such that the function $f_i(\mathbf{p}_j)$ approximates the geographical distance $d_{i,j} = d(S_i, S_j) \in \Re$ from anchor node S_i to the other nodes S_j as closely as possible using proximity $\mathbf{p}_j \in \mathbb{Z}^M$, where $\mathbf{w}_i$ and b_i are the parameters of the nonlinear function (10), and $\boldsymbol{\varphi}(\cdot)$ is a nonlinear basis function. The goal of the training step is to estimate the parameters of the nonlinear function (10), $\mathbf{w}_i$, and b_i. Once the nonlinear function in Eq. (10) is trained, the conversion of the hop count from anchor node S_i to the distance from anchor node S_i is possible. The nonlinear function training is formulated according to the following regularized sparse optimization [31,33]:

$$\underset{\mathbf{w}_i, b_i}{\text{minimize}} \quad \sum_{j=1}^{M} E_\varepsilon(f_i(\mathbf{p}_j) - d_{i,j}) + \frac{1}{2C}\|\mathbf{w}_i\|^2, \tag{11}$$

where $E_\varepsilon(\cdot)$ is an ε-insensitive error function defined as

$$E_\varepsilon(\cdot) = \begin{cases} 0, & \text{if } |f_i(\mathbf{p}_j) - d_{i,j}| < \varepsilon \\ |f_i(\mathbf{p}_j) - d_{i,j}| - \varepsilon, & \text{otherwise} \end{cases}. \tag{12}$$

Eq. (12) indicates that the error of the nonlinear function (10) is tolerated until it is under the given value ε. Problem (11) can then be rewritten as the following equivalent constrained optimization problem:

$$\begin{aligned} &\text{minimize} \quad \sum_{j=1}^{M}\left(\xi_j+\xi_j^*\right)+\tfrac{1}{2C}\|\mathbf{w}_i\|^2 \\ &\text{subject to} \begin{cases} d_{i,j}-f_i(\mathbf{p}_j) \le \varepsilon+\xi_j \\ f_i(\mathbf{p}_j)-d_{i,j} \le \varepsilon+\xi_j^* \\ \xi_j,\xi_j^* \geqslant 0 \end{cases} \end{aligned} \tag{13}$$

by introducing the slack variables $\xi_j \geqslant 0$ and $\xi_j^* \geqslant 0$. In this paper, Active Set Support Vector Regression [18] is applied to the given localization problem since it yields better regression results and solves the problem considerably faster than the simple Support Vector Regression (SVR) algorithm. The details of the ASVR model and the process for solving ASVR are omitted here since they are only important from the perspective of machine learning, not from the perspective of WSN localization. Once the dual optimization problem of ASVR is solved, the regression function $f_i(\mathbf{p})$ can be represented as

$$f_i(\mathbf{p})=\sum_{j=1}^{M}\left(\hat{\alpha}_{i,j}-\hat{\alpha}_{i,j}^*\right)\mathrm{K}(\mathbf{p},\mathbf{p}_j)+b_i, \tag{14}$$

where $\hat{\alpha}_{i,j}$ and $\hat{\alpha}_{i,j}^*$ denote the solutions to the ASVR model, $K\,(\mathbf{p}_j,\mathbf{p}_k) \triangleq \langle \boldsymbol{\varphi}(\mathbf{p}_j),\ \boldsymbol{\varphi}(\mathbf{p}_k)\rangle$ denotes the kernel function, and $b_i=\sum_{j=1}^{M}\left(\hat{\alpha}_{i,j}-\hat{\alpha}_{i,j}^*\right)$. In this paper, the radial basis kernel function is used for simplicity. Each anchor node $S_i \in$ A broadcasts the parameters of its SVR model $\{\hat{\boldsymbol{\alpha}}_i,\hat{\boldsymbol{\alpha}}_i^*,b_i\}$ and its proximity vector $\mathbf{p}_i$ to all of the sensor nodes, where $\hat{\boldsymbol{\alpha}}_i=\left[\hat{\alpha}_{i,1} \quad \cdots \quad \hat{\alpha}_{i,M}\right]^T \in \Re^M$ and $\hat{\boldsymbol{\alpha}}_i^*=\left[\hat{\alpha}_{i,1}^* \quad \cdots \quad \hat{\alpha}_{i,M}^*\right]^T \in \Re^M$. The communication of the SVR step requires MN messages.

3.3.3. Localization step

Each sensor node $S_k \in \Sigma$ starts the localization after receiving M SVR models and proximity vectors, $\{(\hat{\boldsymbol{\alpha}}_1,\hat{\boldsymbol{\alpha}}_1^*,b_1,\mathbf{p}_1),(\hat{\boldsymbol{\alpha}}_2,\hat{\boldsymbol{\alpha}}_2^*,b_2,\mathbf{p}_2),\cdots,(\hat{\boldsymbol{\alpha}}_M,\hat{\boldsymbol{\alpha}}_M^*,b_M,\mathbf{p}_M)\}$ from the anchor nodes. Let

$$\tilde{\mathbf{d}}_k=\left[\tilde{d}_{k,1} \quad \cdots \quad \tilde{d}_{k,M}\right]^T \tag{15}$$

be the estimated distance vector representing the estimated distance between the sensor node S_k and M anchor nodes $S_j \in$ A, where $\tilde{d}_{k,j}$ denotes the estimated distance between $S_k \in \Sigma$ and $S_j \in A$. The estimated distance vector $\tilde{\mathbf{d}}_k$ can be determined by

$$\tilde{\mathbf{d}}_k=\left[\tilde{d}_{k,1} \quad \cdots \quad \tilde{d}_{k,M}\right]^T,=[f_1(\bar{\mathbf{p}}_k) \quad \cdots \quad f_M(\bar{\mathbf{p}}_k)]^T,=\left[\sum_{j=1}^{M}(\hat{\alpha}_{1,j}-\hat{\alpha}_{1,j}^*)\mathrm{K}(\bar{\mathbf{p}}_k,\mathbf{p}_j)+b_1 \quad \cdots \quad \sum_{j=1}^{M}\left(\hat{\alpha}_{M,j}-\hat{\alpha}_{M,j}^*\right)\mathrm{K}(\bar{\mathbf{p}}_k,\mathbf{p}_j)+b_M\right]^T. \tag{16}$$

After the estimates of the distances $\tilde{\mathbf{d}}_k$ to the anchor nodes are evaluated, multilateration is performed to localize the position of the sensor node S_k. Among several multilateration algorithms reported for localization in the WSN [11,19,21], the method used in [11] is used in this paper for the sake of simplicity.

3.4. Localization through Multi-dimensional Support Vector Regression (LMSVR)

The second proposed method is the multi-dimensional version of the kernelized regression approach to the localization, called LMSVR (*Localization through Multi-dimensional Support Vector Regression*). In LSVR, the one-dimensional *distance regression* model is built based on the proximity measurements, and the position of a sensor node $S_k \in \Sigma$ is estimated by applying multilateration to the estimated distances of the anchor nodes. In LMSVR, however, the two-dimensional *location regression* model is trained directly using the proximity measurements and the position of each sensor node is directly estimated simply by evaluating the proposed MSVR model, which will be given shortly. It is assumed that an anchor node is selected as the sink node, which trains the MSVR model, and therefore should be the most resourceful node. The LMSVR is summarized in the following steps:

STEP 1 (Measurement Step): Each anchor node $S_i \in A$ and sensor node $S_j \in \Sigma$ exchanges the hop count information, and each anchor node S_i transmits its position information to a sink node $S_k \in A$, which is one of the anchor nodes.
STEP 2 (MSVR Step): The sink node $S_k \in A$ estimates the location model using the proposed MSVR method and broadcasts it to all sensor nodes $S_j \in \Sigma$.
STEP 3 (Localization Step): Each sensor node $S_j \in \Sigma$ estimates its position using the MSVR model given by the sink node in the previous MSVR step.

The communication costs of the entire proposed LMSVR are $O(MN)$ messages. The details of the proposed LMSVR are as follows. The first step, the measurement step, is the same as that of the LSVR except that every anchor node $S_i \in A$, excluding the sink node S_k, sends an INFO message including its proximity vector $\mathbf{p}_i$ and location information $\mathbf{pos}(S_i)$ to the sink node S_k. The communication costs of the measurement step are $M(M + N - 1)$ HELLO messages and $M - 1$ INFO messages. In the MSVR step, a sink node $S_k \in A$ estimates the location model using the proposed MSVR, and broadcasts it to all sensor nodes.

3.4.1. MSVR step

In the MSVR step, M training data pairs $(\mathbf{p}_i, \mathbf{pos}(S_i)) \in \mathbb{Z}^M \times \Re^2$ $(i = 1, 2, \ldots, M)$ are collected from the anchor nodes $S_i \in A$. Unlike the SVR, a two-dimensional function

$$\mathbf{f}_{MSVR}(\mathbf{p}_i) = \mathbf{W}^T \boldsymbol{\varphi}(\mathbf{p}_i) + \mathbf{b} = \begin{bmatrix} \mathbf{w}_x^T \\ \mathbf{w}_y^T \end{bmatrix} \boldsymbol{\varphi}(\mathbf{p}_i) + \begin{bmatrix} b_x \\ b_y \end{bmatrix} \tag{17}$$

is trained such that the nonlinear function $\mathbf{f}_{MSVR}(\mathbf{p}_i)$ directly approximates the location of node S_i, i.e., $\mathbf{pos}(S_i)$, as closely as possible without resorting to multilateration. The training of $\mathbf{f}_{MSVR}(\mathbf{p}_i)$ is formulated according to the following optimization problem:

$$\text{minimize } L(\mathbf{W}, \mathbf{b}) = \frac{1}{2C}(\|\mathbf{w}_x\|^2 + \|\mathbf{w}_y\|^2) + \sum_{i=1}^{M} L_\varepsilon(u_i), \tag{18}$$

where $u_i = |\mathbf{e}_i|$ and $\mathbf{e}_i = \mathbf{pos}(S_i) - \mathbf{W}^T\boldsymbol{\varphi}(\mathbf{p}_i) - \mathbf{b}$. This training is taken from [25]. In MSVR, the Vapnik ε-insensitive loss function $L_\varepsilon(\cdot)$ is extended to multiple dimensions using the L_2-norm, which is represented by

$$L_\varepsilon(x) = \begin{cases} 0, & x < \varepsilon \\ x^2 - 2x\varepsilon + \varepsilon^2, & x \geqslant \varepsilon \end{cases}. \tag{19}$$

Since the problem cannot be optimized directly, an iterative approach is taken to solve the problem. The details of solving this problem are given in Appendix A. The resulting MSVR model is represented as

$$\mathbf{f}_{MSVR}(\mathbf{p}) = \begin{pmatrix} \sum_{i=1}^{M} \hat{\beta}_x \boldsymbol{\varphi}^T(\mathbf{p})\boldsymbol{\varphi}(\mathbf{p}_i) + \hat{b}_x \\ \sum_{i=1}^{M} \hat{\beta}_y \boldsymbol{\varphi}^T(\mathbf{p})\boldsymbol{\varphi}(\mathbf{p}_i) + \hat{b}_y \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{M} \hat{\beta}_x K(\mathbf{p}, \mathbf{p}_i) + \hat{b}_x \\ \sum_{i=1}^{M} \hat{\beta}_y K(\mathbf{p}, \mathbf{p}_i) + \hat{b}_y \end{pmatrix} = \begin{bmatrix} \hat{\boldsymbol{\beta}}_x^T \\ \hat{\boldsymbol{\beta}}_y^T \end{bmatrix} \mathbf{K}_\mathbf{p} + \begin{bmatrix} \hat{b}_x \\ \hat{b}_y \end{bmatrix} = \hat{\boldsymbol{\beta}}^T \mathbf{K}_\mathbf{p} + \hat{\mathbf{b}}, \tag{20}$$

where $\hat{\boldsymbol{\beta}} = \begin{bmatrix} \hat{\boldsymbol{\beta}}_x & \hat{\boldsymbol{\beta}}_y \end{bmatrix} = \begin{bmatrix} \beta_x & \cdots & \beta_x & \beta_y & \cdots & \beta_y \end{bmatrix}^T \in \Re^{2M}, \hat{\mathbf{b}} = \begin{bmatrix} \hat{b}_x & \hat{\mathbf{b}}_y \end{bmatrix}^T$ represent the solution of (18) using the kernel function. In addition, using the representation theorem [27]:

$$\begin{aligned} \mathbf{w}_x &= \sum_{i=1}^{M} \boldsymbol{\varphi}(\mathbf{p}_i)\beta_x = \boldsymbol{\Phi}^T \boldsymbol{\beta}_x \\ \mathbf{w}_y &= \sum_{i=1}^{M} \boldsymbol{\varphi}(\mathbf{p}_i)\beta_y = \boldsymbol{\Phi}^T \boldsymbol{\beta}_y \end{aligned} \tag{21}$$

and

$$\mathbf{K}_\mathbf{p} = [K(\mathbf{p}, \mathbf{p}_1) \quad K(\mathbf{p}, \mathbf{p}_2) \quad \cdots \quad K(\mathbf{p}, \mathbf{p}_M)]^T \tag{22}$$

is the kernel function evaluated at vector $\mathbf{p}$ and training points $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_M\}$. After the MSVR model is trained, the parameters of the MSVR model $\{\hat{\boldsymbol{\beta}}, \hat{\mathbf{b}}\}$ and the training points $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_M\}$ are broadcasted to all sensor nodes by the sink node. The communication costs of the MSVR step are N messages.

3.4.2. Localization step

Each sensor node $S_i \in \Sigma$ starts this step by receiving the MSVR model $\{\hat{\boldsymbol{\beta}}, \hat{\mathbf{b}}\}$. Each sensor node $S_j \in \Sigma$ estimates its own location by

$$(\tilde{x}_j, \tilde{y}_j)^T = \mathbf{f}_{MSVR}(\bar{\mathbf{p}}_j), \quad = \hat{\boldsymbol{\beta}}^T \boldsymbol{\Phi} \boldsymbol{\varphi}(\bar{\mathbf{p}}_j) + \hat{\mathbf{b}}, \quad = \hat{\boldsymbol{\beta}}^T \mathbf{K}_{\bar{\mathbf{p}}_j} + \hat{\mathbf{b}}, \tag{23}$$

where $\hat{\boldsymbol{\beta}} = \begin{bmatrix} \hat{\boldsymbol{\beta}}_x & \hat{\boldsymbol{\beta}}_y \end{bmatrix}, \hat{\mathbf{b}} = \begin{bmatrix} \hat{b}_x & \hat{b}_y \end{bmatrix}^T$, and $\mathbf{K}_{\bar{\mathbf{p}}_j}$ is the kernel transform of the input vector $\bar{\mathbf{p}}_j$ and training points $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_M\}$.

Remark 1. The SVR does not train a model with the objective of squared error minimization, but by performing structural risk minimization (SRM). Thus, Gaussianity of the error distribution is not assumed in the proposed method, unlike the PDM [15]. The PDM, however, uses the least squares algorithm and requires explicit or implicit assumption of the Gaussianity of the error distribution.

4. Experimental results

In this section, some simulations are conducted to compare the performances of the proposed methods with those of the previous methods including DV-hop [21], PDM [15], and LSVM [32]. The three previous methods require the same conditions as the proposed methods: (1) only connectivity information between all of the nodes and the position of the anchor nodes are used (any other measurements including RSSI, TOA, AOA are not required), and (2) the range of all of the sensor and anchor nodes are identical. DV-hop is one of the most popular range-free algorithms, performing especially well in the case of an isotropic network. PDM considers the topological characteristics of the given network for the localization and performs well, especially in the case of an anisotropic network. LSVM is a recently proposed range-free algorithm that uses a kernelized classification approach.

All simulations are performed on a PC with an Intel 2.4-GHz CPU and 2 GB of memory, and all of the competing methods (DV-hop, PDM, LSVM, LSVR, and LMSVR) are implemented in MATLAB version 7.4. In the simulation, sensor and anchor nodes are distributed in a 100 × 100 square region and two kinds of networks are considered: (1) an isotropic network in which nodes are placed uniformly across the entire network such that the density and connectivity are approximately the same throughout the network and (2) an anisotropic network in which nodes are not placed uniformly. The following probability density function is considered for the construction of the isotropic network:

$$f(x,y) = \begin{cases} \frac{1}{D_x D_y}, & \text{for } 0 \leqslant x \leqslant D_x \text{ and } 0 \leqslant y \leqslant D_y \\ 0, & \text{otherwise} \end{cases} \tag{24}$$

where D_x and D_y are equal to 100.

Figs. 4 and 5 show an example of an isotropic network and anisotropic network, respectively.

Three different simulations are performed.

(a) In the first simulation, three sensor networks with different node densities are considered: (1) a sparse network with 21 anchor nodes and 70 sensor nodes, (2) an intermediate network with 21 anchor nodes and 210 sensor nodes, and (3) a dense network with 100 anchor nodes and 400 sensor nodes. The communication range R for all nodes is assumed to be constant.

(b) In the second simulation, the network size is fixed to 300, and the anchor node ratio is changed from 10% to 30% to determine the variation in performance. The communication range is assumed to be constant.

(c) In the third simulation, more realistic environments are considered. The communication range R for all nodes is set to a modest value, and an irregular radio propagation model is used. The network size is fixed to 300 and the anchor node ratio is changed from 10% to 30% to determine the variation in performance.

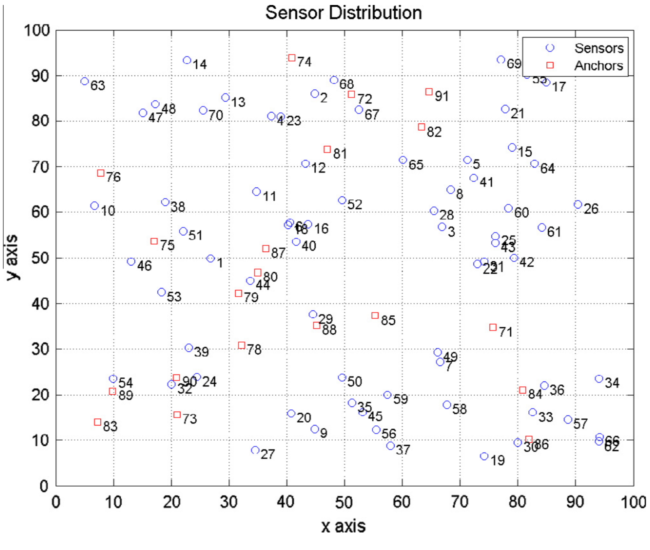


Fig. 4. An example of an isotropic network topology used in the simulation.

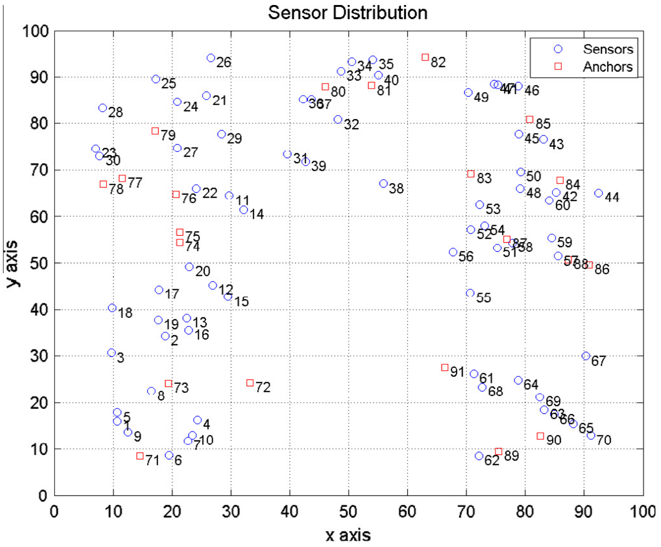


Fig. 5. An example of an anisotropic network topology used in the simulation.

Table 1
Location error of the proposed methods with various parameters in an isotropic network.

Location error (m)		$\sigma = 1$	$\sigma = 3$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$
$C = 1$	LSVR	36.7840	21.8985	14.7587	13.5343	22.0114
	LMSVR	34.1674	16.5540	10.5938	10.3510	17.5327
$C = 3$	LSVR	34.3036	16.2992	9.5457	8.7420	12.0324
	LMSVR	32.4767	13.6299	8.0344	7.7642	10.3735
$C = 5$	LSVR	35.2121	15.1586	9.9385	8.5101	11.0726
	LMSVR	33.4321	13.5394	9.1416	7.9216	10.0105
$C = 10$	LSVR	33.5963	13.5518	8.8258	8.2621	9.5717
	LMSVR	31.9531	12.2792	8.3243	7.9654	9.5470

Before starting the three simulations described above, it is worth noting that many kernel-based methods, including SVR, often have the problem of overfitting phenomena, and selection of the design parameters is important in order to determine the trade-off between specificity and sensitivity. In the proposed LSVR and LMSVR, the regularization parameter C in (13) and (18) controls the tradeoff between the model complexity and approximation accuracy in order to ensure good generalization performance [3]. The kernel function $\mathrm{K}(\mathbf{p}_j, \mathbf{p}_k) \triangleq \langle \boldsymbol{\varphi}(\mathbf{p}_j), \boldsymbol{\varphi}(\mathbf{p}_k) \rangle$ also plays an important role in determining the smoothness properties of the SVR and MSVR model, i.e., (14) and (20). In this paper, the following Gaussian kernel is used:

$$\mathrm{K}(\mathbf{p}_j, \mathbf{p}_k) = \exp\left(-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\sigma^2}\right), \tag{25}$$

where σ is a constant. The choice of σ also plays a crucial role in determining the trade-off between specificity and sensitivity. Therefore, simulations are conducted to properly determine the parameters C and σ in order to avoid the overfitting phenomena. Tables 1 and 2 show the average localization error of 70 sensor nodes with 21 training anchor nodes for various values of C and σ. Note that ten independent simulations are performed for each case. From the tables, it is evident that the overfitting phenomena do not occur and the proposed methods work well when C and σ are set to be 10. Therefore, we set C and σ to be 10 in simulations (a)–(c).

In simulation (a), the communication range R for all nodes is set to 20. For each topology, ten independent simulation runs are made. Table 3 compares the performances of the isotropic network, while Table 4 compares the performances of the anisotropic network. From the tables, it is apparent that the proposed LSVR and LMSVR outperform the previous methods not only on average, but also in the best and worst cases for both the isotropic and anisotropic networks. In the isotropic network, as the network gets denser, all of the competing methods demonstrate a performance improvement, except the PDM method. In the anisotropic network, however, as the network gets denser, only the proposed LSVR and LMSVR exhibit a substantial performance improvement. The previous methods do not exhibit satisfactory improvement.

Table 2
Location error of the proposed methods with various parameters in an anisotropic network.

Location error (m)		$\sigma = 1$	$\sigma = 3$	$\sigma = 5$	$\sigma = 10$	$\sigma = 20$
$C = 1$	LSVR	33.3129	17.5209	11.5969	12.4551	17.4839
	LMSVR	30.8802	13.3758	8.8716	10.1006	14.1418
$C = 3$	LSVR	31.3008	11.8941	8.7073	9.5766	11.7754
	LMSVR	29.4112	10.2671	7.8046	8.4982	10.3676
$C = 5$	LSVR	31.1916	10.7232	8.2739	8.3446	10.3842
	LMSVR	29.9873	9.7675	7.7458	7.5598	9.1473
$C = 10$	LSVR	29.0866	10.3503	7.8093	7.4843	9.2755
	LMSVR	28.5315	9.7439	7.5074	7.0699	8.4302

Table 3
Comparison of the location error for the isotropic network.

Location error (m)		DV-hop	PDM	LSVM	LSVR	LMSVR
21 Anchor, 70 sensor	Avg.	9.2856	26.5096	20.1761	6.7242	6.6811
	Best	7.4082	11.0391	16.4354	3.6754	3.6246
	Worst	12.3142	63.1934	21.9665	9.2863	9.0827
21 Anchor, 210 sensor	Avg.	7.4385	24.7650	20.4885	6.2708	6.1854
	Best	4.9420	15.5129	17.9729	4.3275	4.6977
	Worst	9.5887	37.9226	23.3557	9.3838	8.8882
100 Anchor, 400 sensor	Avg.	5.2703	40.3711	14.9135	2.1454	1.9131
	Best	3.9580	15.2745	12.0408	1.6187	1.4640
	Worst	6.5874	115.2371	16.2604	2.7872	2.4305

Table 4
Comparison of the location error for the anisotropic network.

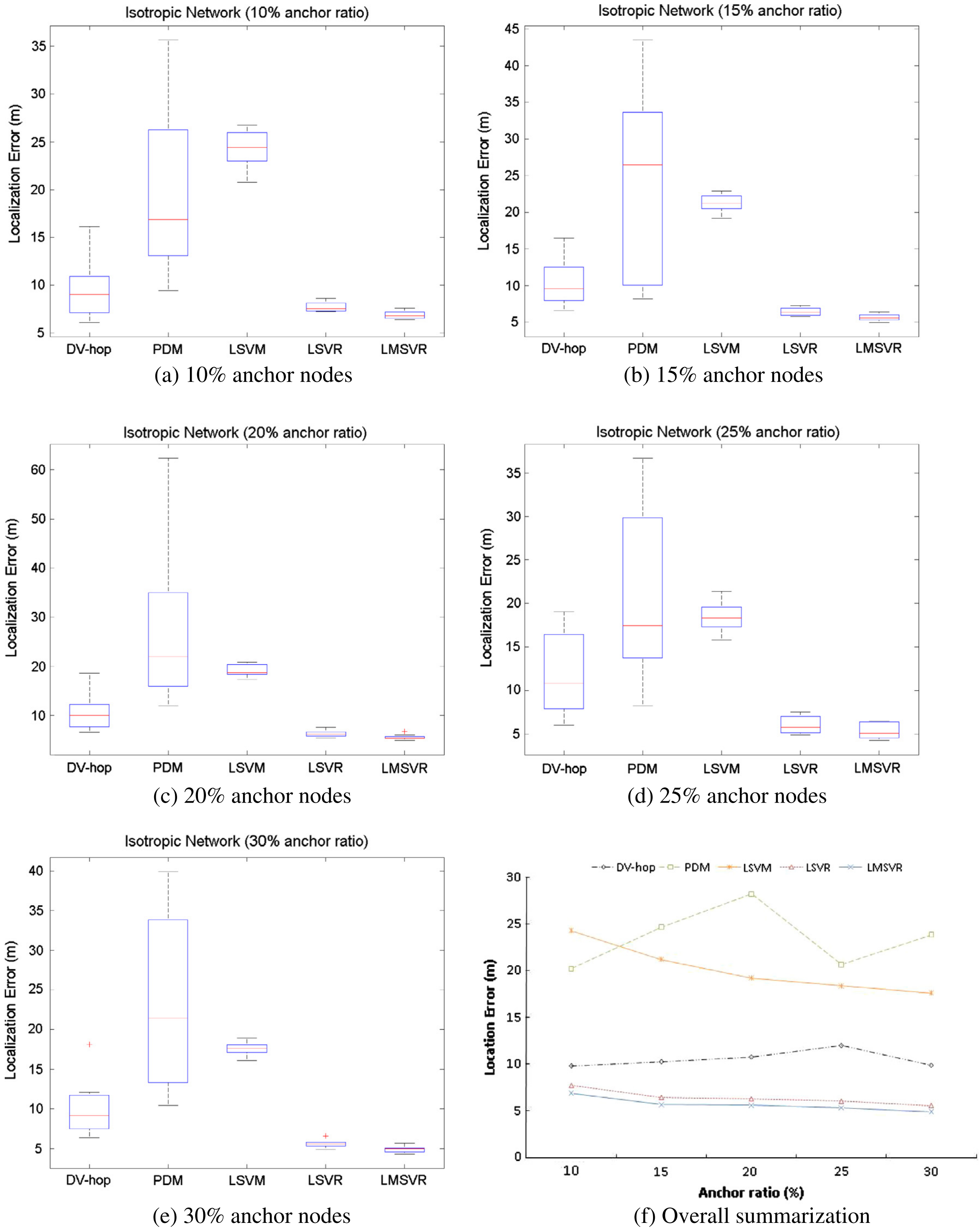
Location error (m)		DV-hop	PDM	LSVM	LSVR	LMSVR
21 Anchor, 70 sensor	Avg.	33.3337	12.8764	19.3585	7.6265	7.1298
	Best	19.4225	7.1781	17.7222	6.0780	5.7668
	Worst	50.1211	24.7856	21.4206	8.8597	7.8337
21 Anchor, 210 sensor	Avg.	25.6624	19.7441	19.2662	7.1949	6.6708
	Best	21.3248	9.8345	17.2365	5.7152	5.5767
	Worst	34.0291	45.6732	20.5723	8.5571	7.9714
100 Anchor, 400 sensor	Avg.	25.6513	21.1240	15.8426	3.0213	2.7262
	Best	18.3951	10.9769	15.0911	2.6768	2.4368
	Worst	29.4402	37.2566	16.5378	3.3049	2.9754

Table 5
Comparison of the average computation time for two networks.

CPU time (s)		DV-hop	PDM	LSVM	LSVR	LMSVR
Isotropic network	21 Anchors, 70 sensors	0.0125	0.0182	0.0151	0.0088	0.0735
	21 Anchors 210 sensors	0.0161	0.0208	0.0125	0.0145	0.0469
	100 Anchors 400 sensors	0.0365	0.0677	0.1109	0.0792	53.8391
Anisotropic network	21 Anchors, 70 sensors	0.0156	0.0114	0.0146	0.0093	0.0391
	21 Anchors 210 sensors	0.0125	0.0234	0.0146	0.0142	0.0375
	100 Anchors 400 sensors	0.0365	0.0615	0.1188	0.0775	38.9734

The dense network has the least localization error irrespective of the localization method, and there is a clear trade-off between the node density and localization performance. However, considering the saturation of the localization accuracy stated above, the network having an unnecessarily high density should be avoided.

Furthermore, the proposed method is compared with the previous methods in terms of complexity (computation time), given in Table 5. The proposed LMSVR method takes more time than the other methods, especially in the case of a dense



(a) 10% anchor nodes

(b) 15% anchor nodes

(c) 20% anchor nodes

(d) 25% anchor nodes

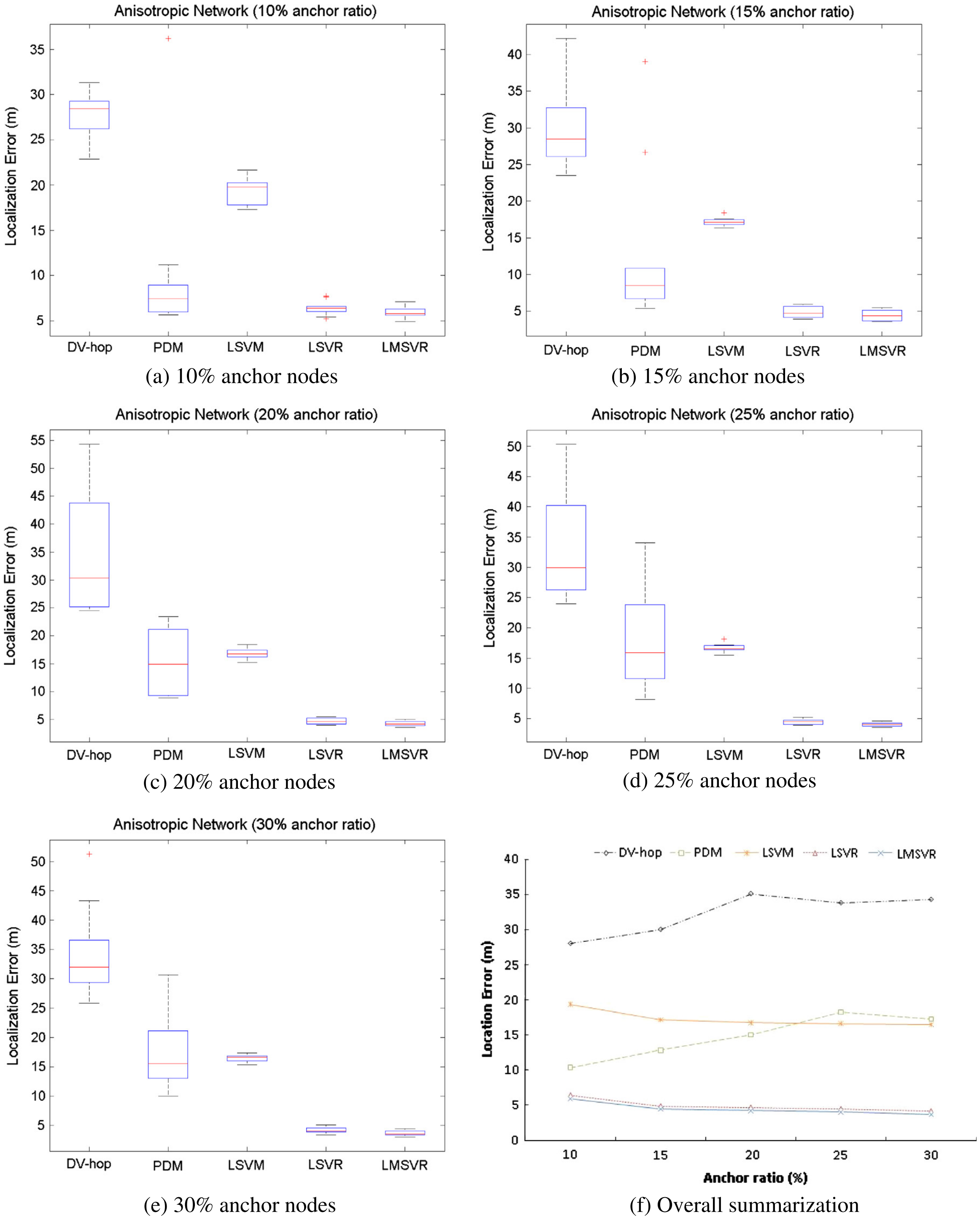
(e) 30% anchor nodes

(f) Overall summarization

Fig. 6. Comparison of the location error for the isotropic network under various anchor populations.

network. This could be due to the rapid increase in the computation time of the MSVR as the training sets or the number of anchor nodes increase.

In simulation (b), the number of nodes is fixed at 300, and the anchor ratio is varied from 10% to 30%. The communication range R is set to 10 for all nodes. Localization performances are compared for the isotropic and anisotropic networks using box plots in Figs. 6 and 7, respectively. The box plots demonstrate the minimum and maximum of the localization error, the



(a) 10% anchor nodes

(b) 15% anchor nodes

(c) 20% anchor nodes

(d) 25% anchor nodes

(e) 30% anchor nodes

(f) Overall summarization

Fig. 7. Comparison of the location error for the anisotropic network under various anchor populations.

median, and the 25th and 75th percentiles of ten independent simulations. Figures (a)–(e) show the localization results of the network in which the anchor ratio is 10%, 15%, 20%, 25%, and 30%, respectively. The boxes represent the results of the DV-hop, PDM, LSVM, LSVR, and LMSVR, in that order.

From the figures, it is evident that the localization accuracy depends proportionally on the number of anchor nodes for the kernel-based localization algorithms, LSVM, LSVR, and LMSVR. However, when the number of anchor nodes is very small

Table 6
Comparison of the computation time with various anchor ratios.

Anchor ratio		DV-hop	PDM	LSVM	LSVR	LMSVR
Isotropic network	10%	0.0318	0.0224	0.0187	0.0192	0.7234
	15%	0.0266	0.0234	0.0438	0.0249	9.1265
	20%	0.0198	0.0307	0.0734	0.0315	18.1202
	25%	0.0255	0.0365	0.1188	0.0424	25.2928
	30%	0.0266	0.0422	0.1859	0.0559	39.4812
Anisotropic network	10%	0.0239	0.0208	0.0250	0.0192	0.6672
	15%	0.0219	0.0297	0.0453	0.0251	2.9281
	20%	0.0255	0.0271	0.0797	0.0311	13.4681
	25%	0.0265	0.0328	0.1266	0.0442	15.3437
	30%	0.0261	0.0401	0.1813	0.0557	43.0984

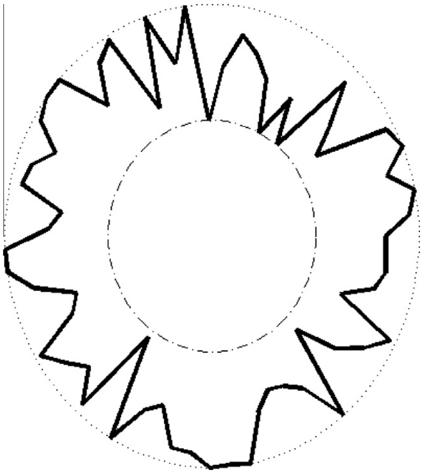


Fig. 8. Radio propagation model.

(when the anchor ratio is only 10%), the proposed methods demonstrate much better performance than the previous methods. More specifically, as in the first simulation, the proposed LSVR and LMSVR outperform the previous methods on average, demonstrating more consistent performance than the others for both isotropic and anisotropic networks. In particular, LSVR and LMSVR exhibit a performance improvement as the number of anchor nodes increases. This observation makes sense because as the number of anchor nodes increases, more information becomes available for the training of SVR and MSVR. The DV-hop method shows good performance in the case of an isotropic network, but it does not in the case of an anisotropic network. The for this reason is that the DV-hop method assumes that a large hop count means a long distance, but this is rarely the case in an anisotropic network. By contrast, the PDM method exhibits a reasonable performance in the case of an anisotropic network since it considers the characteristics of the given network for localization.

Furthermore, the proposed methods are compared with the previous methods in terms of their complexity (computation time), and the results are summarized in Table 6. It is apparent that the DV-hop and PDM methods have consistent times regardless of changes in their anchor ratio, while the computation times of kernel-based approaches, including LSVM, LSVR, and LMSVR, increase with the anchor ratio.

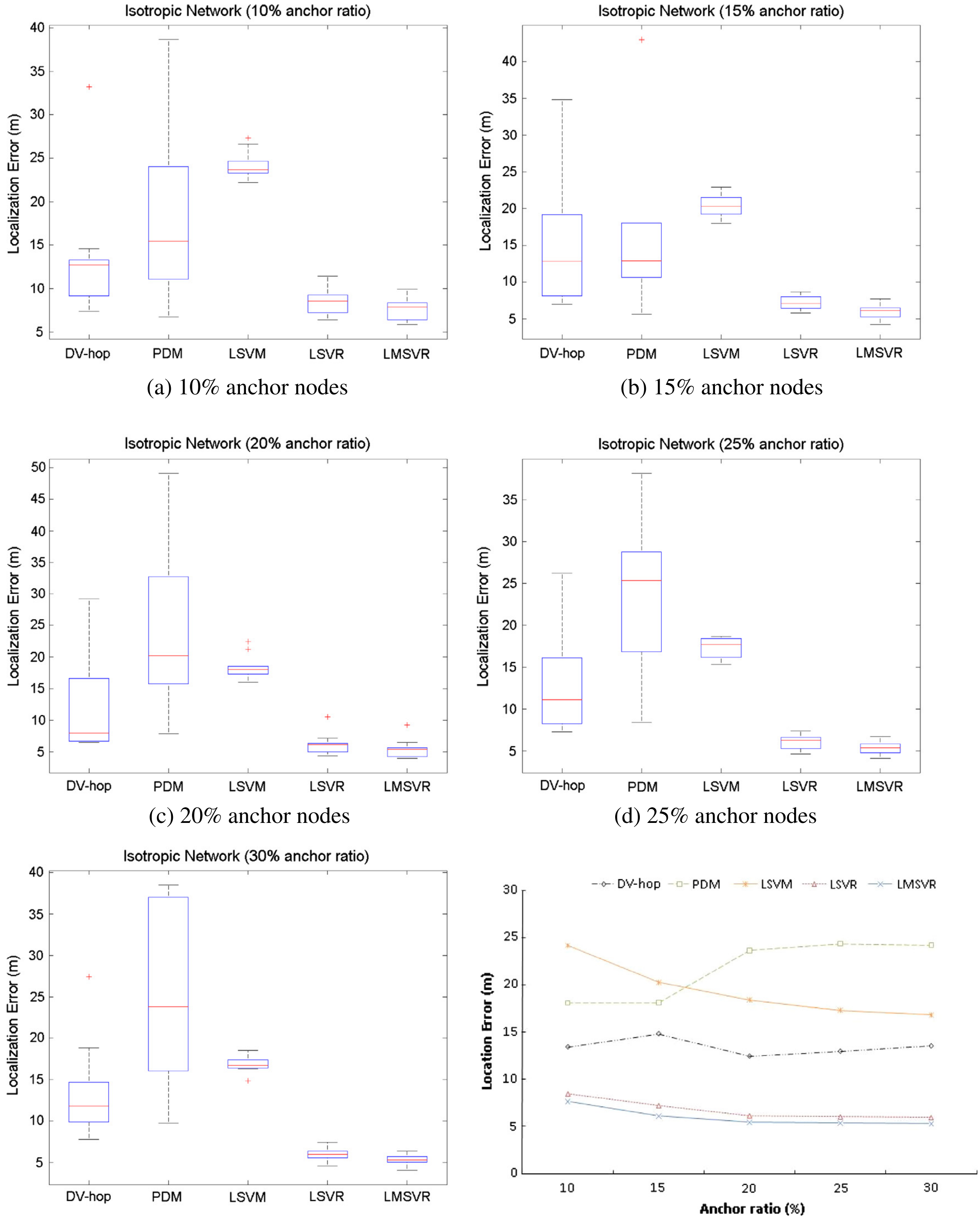
In simulation (c), a more realistic environment is considered. The irregular radio propagation model taken from [29] is employed to model the fluctuation of communication range R due to the multipath channel, interference, and other effects in the real environment. The irregularity is represented by the *DOI* (degree of irregularity), and the communication range is represented by

$$(1 - DOI)R \leqslant \text{communication range} \leqslant R. \tag{26}$$

Fig. 8 shows an example of the radio propagation model.

The total network size is fixed to 300, and the anchor ratio varies from 10% to 30%. The communication range R is set to 10, and the *DOI* value is assumed to be 0.2. The localization performances for isotropic and anisotropic networks are compared in box plots in Figs. 9 and 10, respectively. The box plots represent the statistics of ten independent runs.

As in the above two simulations, the proposed LSVR and LMSVR demonstrate better and more consistent performance than the previous methods, as indicated by the box plots. For both isotropic and anisotropic networks, LSVR and LMSVR



(a) 10% anchor nodes

(b) 15% anchor nodes

(c) 20% anchor nodes

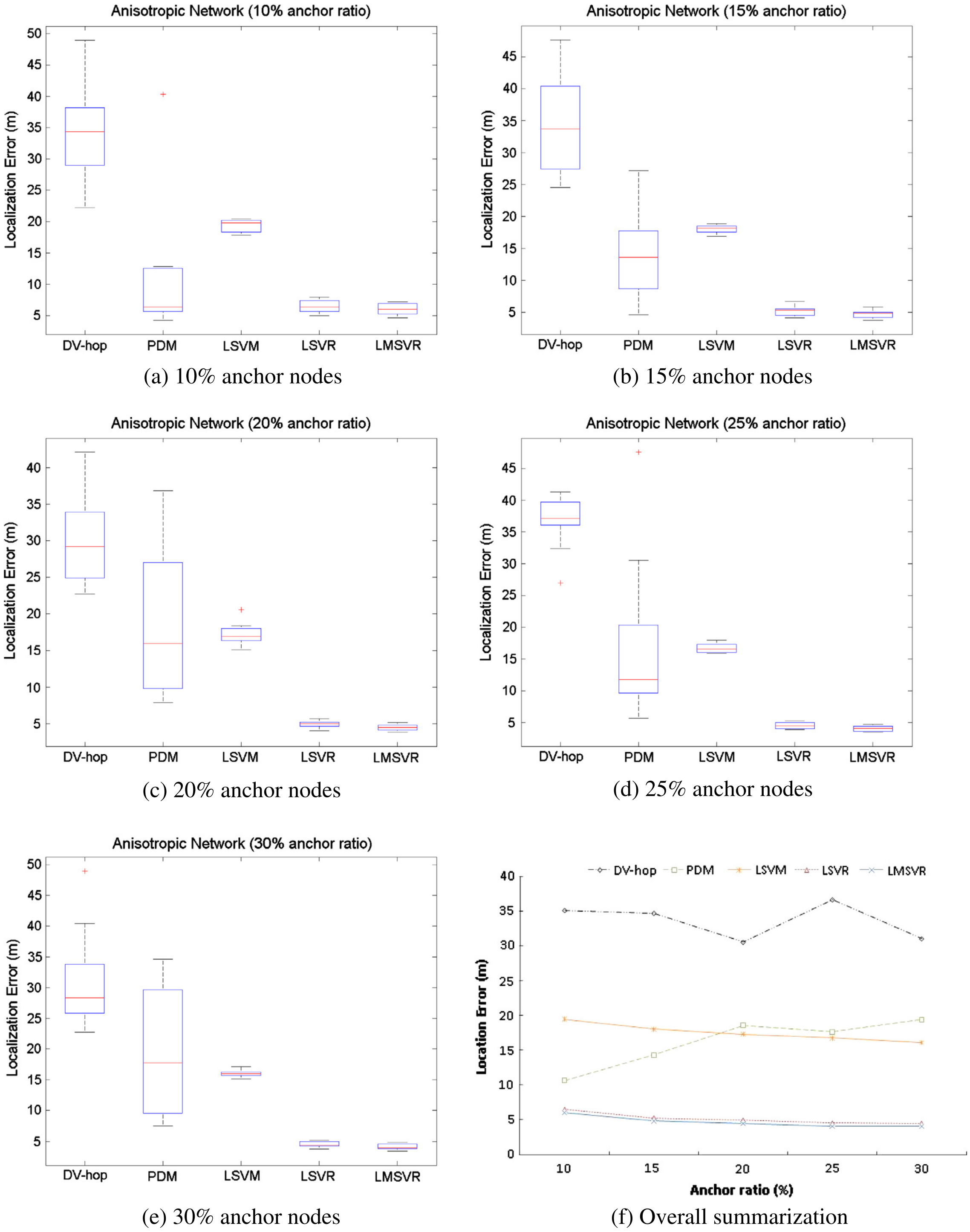
(d) 25% anchor nodes

(e) 30% anchor nodes

(f) Overall summarization

Fig. 9. Comparison of the location error for the isotropic network under various anchor populations with noisy radio propagation.

exhibit improved localization performance as the number of anchor nodes increases. It is worth noting that the localization errors of the proposed LSVR and LMSVR converge as the anchor ratio increases. From the perspective of machine learning, this phenomenon can be explained by the sparsity of the SVR. Even though the number of training samples increases, most of the samples are ignored, and only a limited number of important samples contribute to the SVR, thereby causing the



(a) 10% anchor nodes

(b) 15% anchor nodes

(c) 20% anchor nodes

(d) 25% anchor nodes

(e) 30% anchor nodes

(f) Overall summarization

Fig. 10. Comparison of the location error for the anisotropic network under various anchor populations with noisy radio propagation.

convergence of the localization error. The important samples are the "support vectors" in the SVR model, and the ignored training samples sparsify the resulting SVR.

Finally, to determine the reliability of the above three simulations, non-parametric statistical tests are executed. The Wilcoxon rank sum test is used to compare the proposed methods, LSVR and LMSVR, and the previous methods, DV-hop, PDM, and LSVM. Tables 7–9 show the p-values of the test results.

Table 7
Statistical test results for simulation (A).

Wilcoxon rank sum test (*p*-value)			DV-hop	PDM	LSVM
Isotropic network	21 Sensor nodes	LSVR	0.0028	1.8267e−04	1.8267e−04
	70 Anchor nodes	LMSVR	0.0028	1.8267e−04	1.8267e−04
	21 Sensor nodes	LSVR	0.1041	1.8267e−04	1.8267e−04
	210 Anchor nodes	LMSVR	0.0890	1.8267e−04	1.8267e−04
	100 Sensor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
	400 Anchor nodes	LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
Anisotropic network	21 Sensor nodes	LSVR	1.8267e−04	0.0091	1.8267e−04
	70 Anchor nodes	LMSVR	1.8267e−04	0.0017	1.8267e−04
	21 Sensor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
	210 Anchor nodes	LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
	100 Sensor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
	400 Anchor nodes	LMSVR	1.8267e−04	1.8267e−04	1.8267e−04

Table 8
Statistical test results for simulation (B).

Wilcoxon rank sum test (*p*-value)			DV-hop	PDM	LSVM
Isotropic network	10% Anchor nodes	LSVR	0.1620	1.8267e−04	1.8267e−04
		LMSVR	0.0257	1.8267e−04	1.8267e−04
	15% Anchor nodes	LSVR	7.6854e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
	20% Anchor nodes	LSVR	0.0010	1.8267e−04	1.8267e−04
		LMSVR	3.2984e−04	1.8267e−04	1.8267e−04
	25% Anchor nodes	LSVR	0.0022	1.8267e−04	1.8267e−04
		LMSVR	5.8284e−04	1.8267e−04	1.8267e−04
	30% Anchor nodes	LSVR	2.4613e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
Anisotropic network	10% Anchor nodes	LSVR	1.8267e−04	0.1405	1.8267e−04
		LMSVR	1.8267e−04	0.0140	1.8267e−04
	15% Anchor nodes	LSVR	1.8267e−04	4.3964e−04	1.8267e−04
		LMSVR	1.8267e−04	2.4613e−04	1.8267e−04
	20% Anchor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
	25% Anchor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
	30% Anchor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04

Table 9
Statistical test results for simulation (C).

Wilcoxon rank sum test (*p*-value)			DV-hop	PDM	LSVM
Isotropic network	10% Anchor nodes	LSVR	0.0140	0.0036	1.8267e−04
		LMSVR	0.0046	0.0013	1.8267e−04
	15% Anchor nodes	LSVR	0.0036	0.0028	1.8267e−04
		LMSVR	3.2984e−04	0.0013	1.8267e−04
	20% Anchor nodes	LSVR	0.0028	3.2984e−04	1.8267e−04
		LMSVR	0.0013	2.4613e−04	1.8267e−04
	25% Anchor nodes	LSVR	2.4613e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
	30% Anchor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
Anisotropic network	10% Anchor nodes	LSVR	1.8267e−04	0.7337	1.8267e−04
		LMSVR	1.8267e−04	0.4274	1.8267e−04
	15% Anchor nodes	LSVR	1.8267e−04	0.0022	1.8267e−04
		LMSVR	1.8267e−04	0.0017	1.8267e−04
	20% Anchor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
	25% Anchor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04
	30% Anchor nodes	LSVR	1.8267e−04	1.8267e−04	1.8267e−04
		LMSVR	1.8267e−04	1.8267e−04	1.8267e−04

From the tables, it is evident that the superiority of LSVR and LMSVR over the DV-hop, PDM, and LSVM methods is significant for almost all simulations. The p-values between the proposed and previous methods are very low for most of the simulations, which means that the differences in performance between the methods being compared are significant. In the case of an isotropic network, the largest p-value is found between LSVR and DV-hop in simulation (b), with a value of 0.1620. In the case of an anisotropic network, the largest p-value is found between LSVR and PDM in simulation (c), with a value of 0.7337.

5. Conclusions

In this paper, a new kernelized regression approach to localization in WSNs was proposed. The localization problem was formulated as a kernelized regression problem, and the solutions were proposed using nonlinear mapping and the introduction of a kernel function. Only simple proximity information among the sensors was used as measurements, and each sensor node estimated its own location in a distributed manner. Simulations were conducted to compare the proposed methods with the previous methods under a variety of conditions including both isotropic and anisotropic networks, and various anchor populations and communication ranges with interference and noise. Compared with the previous methods, it was observed that the proposed methods exhibit better performance in terms of the average localization error and its consistency.

Acknowledgment

This work was supported by "Cognitive model-based global localization for indoor robots" (Project number: 10031687) of the Ministry of Knowledge Economy, Republic of Korea.

Appendix A

The subsequent derivation is almost the same as in [25], but it is given here for completeness. Consider the problem in Eq. (17). To construct the iterative procedure, $L_\varepsilon(\cdot)$ in Eq. (18) is linearized by the first-order Taylor expansion, and Eq. (18) can be rewritten as

$$\widetilde{L}(\mathbf{W},\mathbf{b}) = \frac{1}{2}(\|\mathbf{w}_x\|^2 + \|\mathbf{w}_y\|^2) + C\left(\sum_{i=1}^{M} L_\varepsilon(u_i^k) + \frac{dL_\varepsilon(u)}{du}\bigg|_{u_i^k} \frac{(\mathbf{e}_i^k)^T}{u_i^k}\left[\mathbf{e}_i - \mathbf{e}_i^k\right]\right), \tag{27}$$

where $u_i^k = \|\mathbf{e}_i^k\|$ and $\mathbf{e}_i^k = \mathbf{pos}(S_i) - (\mathbf{W}^k)^T\boldsymbol{\varphi}(\mathbf{p}_i) - \mathbf{b}^k$, while k denotes the time step of the iteration process. Formulating a quadratic approximation from Eq. (27) yields

$$\begin{aligned}\widetilde{L}_a(\mathbf{W},\mathbf{b}) &= \frac{1}{2}(\|\mathbf{w}_x\|^2 + \|\mathbf{w}_y\|^2) + C\left(\sum_{i=1}^{M} L_\varepsilon(u_i^k) + \frac{dL_\varepsilon(u)}{du}\bigg|_{u_i^k} \frac{u_i^2 - (u_i^k)^2}{2u_i^k}\right)\\ &= \frac{1}{2}(\|\mathbf{w}_x\|^2 + \|\mathbf{w}_y\|^2) + \frac{1}{2}\sum_{i=1}^{M} a_i u_i^2 + \text{constant},\end{aligned} \tag{28}$$

where

$$a_i = \frac{C}{u_i^k}\frac{dL_\varepsilon(u)}{du}\bigg|_{u_i^k} = \begin{cases} 0, & u_i^k < \varepsilon \\ \frac{2C(u_i^k - \varepsilon)}{u_i^k}, & u_i^k \geqslant \varepsilon \end{cases}. \tag{29}$$

The above problem is optimized by

$$\begin{aligned}\nabla_{w_x}\widetilde{L}_a(\mathbf{W},\mathbf{b}) &= \mathbf{w}_x - \sum_{i=1}^{M}\boldsymbol{\varphi}(\mathbf{p}_i)a_i(x_i - \boldsymbol{\varphi}^T(\mathbf{p}_i)\mathbf{w}_x - b_x) = 0,\\ \nabla_{w_y}\widetilde{L}_a(\mathbf{W},\mathbf{b}) &= \mathbf{w}_y - \sum_{i=1}^{M}\boldsymbol{\varphi}(\mathbf{p}_i)a_i(y_i - \boldsymbol{\varphi}^T(\mathbf{p}_i)\mathbf{w}_y - b_y) = 0,\\ \nabla_{b_x}\widetilde{L}_a(\mathbf{W},\mathbf{b}) &= -\sum_{i=1}^{M}a_i(x_i - \boldsymbol{\varphi}^T(\mathbf{p}_i)\mathbf{w}_x - b_x) = 0,\\ \nabla_{b_y}\widetilde{L}_a(\mathbf{W},\mathbf{b}) &= -\sum_{i=1}^{M}a_i(y_i - \boldsymbol{\varphi}^T(\mathbf{p}_i)\mathbf{w}_y - b_y) = 0,\end{aligned} \tag{30}$$

where $\mathbf{pos}(S_i) = (x_i, y_i)^T$. The above equations can be written as the following matrix equation:

$$\begin{bmatrix} \boldsymbol{\Phi}^T\mathbf{D}_a\boldsymbol{\Phi}+\mathbf{I} & \boldsymbol{\Phi}^T\mathbf{a} \\ \mathbf{a}^T\boldsymbol{\Phi} & \mathbf{1}^T\mathbf{a} \end{bmatrix} \begin{bmatrix} \mathbf{w}_x & \mathbf{w}_y \\ b_x & b_y \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}^T\mathbf{D}_a\mathbf{z}_x & \boldsymbol{\Phi}^T\mathbf{D}_a\mathbf{z}_y \\ \mathbf{a}^T\mathbf{z}_x & \mathbf{a}^T\mathbf{z}_y \end{bmatrix}, \tag{31}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\varphi}(\mathbf{p}_1) \quad \cdots \quad \boldsymbol{\varphi}(\mathbf{p}_M)]^T$, $\mathbf{a} = [a_1 \quad \cdots \quad a_M]^T$, $(\mathbf{D}_a)_{i,j} = a_i\delta(i-j)$, $\mathbf{z}_x = [x_1 \quad \cdots \quad x_M]^T$, $\mathbf{z}_y = [y_1 \quad \cdots \quad y_M]^T$, and $\delta(i-j)$ is the Kronecker delta function. The direct nonlinear transformation in Eq. (31) is avoided by applying the kernel function and using the representation theorem [27] as follows:

$$\begin{aligned} \mathbf{w}_x &= \sum_{i=1}^{M}\boldsymbol{\varphi}(\mathbf{p}_i)\beta_x = \boldsymbol{\Phi}^T\boldsymbol{\beta}_x \\ \mathbf{w}_y &= \sum_{i=1}^{M}\boldsymbol{\varphi}(\mathbf{p}_i)\beta_y = \boldsymbol{\Phi}^T\boldsymbol{\beta}_y, \end{aligned} \tag{32}$$

where $\boldsymbol{\beta}_x = [\beta_x \quad \cdots \quad \beta_x]^T \in \Re^M$ and $\boldsymbol{\beta}_y = [\beta_y \quad \cdots \quad \beta_y]^T \in \Re^M$. Substituting Eq. (32) into Eq. (31) yields

$$\begin{bmatrix} \boldsymbol{\Phi}^T\mathbf{D}_a\boldsymbol{\Phi}\boldsymbol{\Phi}^T+\boldsymbol{\Phi}^T & \boldsymbol{\Phi}^T\mathbf{a} \\ \mathbf{a}^T\boldsymbol{\Phi}\boldsymbol{\Phi}^T & \mathbf{1}^T\mathbf{a} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}_x & \boldsymbol{\beta}_y \\ b_x & b_y \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}^T\mathbf{D}_a\mathbf{z}_x & \boldsymbol{\Phi}^T\mathbf{D}_a\mathbf{z}_y \\ \mathbf{a}^T\mathbf{z}_x & \mathbf{a}^T\mathbf{z}_y \end{bmatrix}. \tag{33}$$

By defining a kernel matrix $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$, the following linear system of equations is obtained:

$$\begin{bmatrix} \boldsymbol{\Phi}^T\mathbf{D}_a\mathbf{K}+\boldsymbol{\Phi}^T & \boldsymbol{\Phi}^T\mathbf{a} \\ \mathbf{a}^T\mathbf{K} & \mathbf{1}^T\mathbf{a} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}_x & \boldsymbol{\beta}_y \\ b_x & b_y \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}^T\mathbf{D}_a\mathbf{z}_x & \boldsymbol{\Phi}^T\mathbf{D}_a\mathbf{z}_y \\ \mathbf{a}^T\mathbf{z}_x & \mathbf{a}^T\mathbf{z}_y \end{bmatrix}, \tag{34}$$

Pre-multiplying by

$$\begin{bmatrix} \mathbf{K}^{-1}\boldsymbol{\Phi} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} (\boldsymbol{\Phi}\boldsymbol{\Phi}^T)^{-1}\boldsymbol{\Phi} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \tag{35}$$

on both sides yields

$$\begin{bmatrix} \mathbf{D}_a\mathbf{K}+\mathbf{I} & \mathbf{a} \\ \mathbf{a}^T\mathbf{K} & \mathbf{1}^T\mathbf{a} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}_x & \boldsymbol{\beta}_y \\ b_x & b_y \end{bmatrix} = \begin{bmatrix} \mathbf{D}_a\mathbf{z}_x & \mathbf{D}_a\mathbf{z}_y \\ \mathbf{a}^T\mathbf{z}_x & \mathbf{a}^T\mathbf{z}_y \end{bmatrix}, \tag{36}$$

where $\mathbf{K}$ is a kernel matrix. The radial basis function

$$(\mathbf{K})_{i,j} = \exp\left(-\frac{\|\mathbf{p}_i-\mathbf{p}_j\|^2}{2\sigma^2}\right) \tag{37}$$

is employed for the sake of empirical effectiveness. By alternating between solving Eq. (36) and evaluating Eq. (37), the solutions $\hat{\boldsymbol{\beta}}_x, \hat{\boldsymbol{\beta}}_y, \hat{b}_x$, and $\hat{b}_y$ are obtained, and the MSVR model is represented as

$$\mathbf{f}_{MSVR}(\mathbf{p}) = \begin{pmatrix} \sum_{i=1}^{M}\hat{\beta}_x\boldsymbol{\varphi}^T(\mathbf{p})\boldsymbol{\varphi}(\mathbf{p}_i)+\hat{b}_x \\ \sum_{i=1}^{M}\hat{\beta}_y\boldsymbol{\varphi}^T(\mathbf{p})\boldsymbol{\varphi}(\mathbf{p}_i)+\hat{b}_y \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{M}\hat{\beta}_xK(\mathbf{p},\mathbf{p}_i)+\hat{b}_x \\ \sum_{i=1}^{M}\hat{\beta}_yK(\mathbf{p},\mathbf{p}_i)+\hat{b}_y \end{pmatrix} = \begin{bmatrix} \hat{\boldsymbol{\beta}}_x^T \\ \hat{\boldsymbol{\beta}}_y^T \end{bmatrix}\mathbf{K}_\mathbf{p} + \begin{bmatrix} \hat{b}_x \\ \hat{b}_y \end{bmatrix} = \hat{\boldsymbol{\beta}}^T\mathbf{K}_\mathbf{p} + \hat{\mathbf{b}}, \tag{38}$$

where $\hat{\boldsymbol{\beta}} = [\hat{\boldsymbol{\beta}}_x \quad \hat{\boldsymbol{\beta}}_y]$, $\hat{\mathbf{b}} = \begin{bmatrix} \hat{b}_x & \hat{b}_y \end{bmatrix}^T$, and

$$\mathbf{K}_p = [K(\mathbf{p},\mathbf{p}_1) \quad K(\mathbf{p},\mathbf{p}_2) \quad \cdots \quad K(\mathbf{p},\mathbf{p}_M)]^T \tag{39}$$

is the kernel function evaluated at vector $\mathbf{p}$ and training points $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_M\}$.

References

[1] M. Brunato, R. Battiti, Statistical learning theory for location fingerprinting in wireless LANs, Computer Networks 47 (6) (2005) 825–845.

[2] N. Bulusu, J. Heidemann, D. Estrin, Adaptive beacon placement, in: Proceedings of IEEE International Conference on Distributed Computing Systems, 2001, pp. 489–498.

[3] C.C. Chuang, S.F. Su, J.T. Jeng, C.C. Hsiao, Robust support vector regression networks for function approximation with outliers, IEEE Transactions on Neural Networks 13 (6) (2002) 1322–1330.

[4] Y.-C. Chung, I.-F. Su, C. Lee, An efficient mechanism for processing similarity search queries in sensor networks, Information Sciences 181 (2) (2011) 284–307.

[5] T. He, C. Huang, B.M. Blum, J.A. Stankovic, T. Abdelzaher, Range-free localization schemes for large scale sensor networks, in: Proceedings of the International Conference on Mobile Computing and Networking, 2003, pp. 81–95.

[6] B. Hofmann-Wellenhof, H. Lichtenegger, J. Collins, Global Positioning System: Theory and Practice, Springer-Verlag, 1993.

[7] P. Honeine, C. Richard, M. Essoloh, H. Snoussi, Localization in sensor networks—A matrix regression approach, in: Proceedings pf 5th IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM), 2008, pp. 284–287.

[8] R. Huang, G.V. Zaruba, Incorporating data from multiple sensors for localizing nodes in mobile ad hoc networks, IEEE Transactions on Mobile Computing 6 (9) (2007) 1090–1104.

[9] A.M. Khedr, W. Osamy, Minimum perimeter coverage of query regions in a heterogeneous wireless sensor network, Information Sciences 181 (15) (2011) 3130–3142.

[10] A. Kuh, C. Zhu, D. Mandic, Sensor network localization using least squares kernel regression, in: B. Gabrys, R.J. Howlett, L.C. Jain (Eds.), Proceedings of KES 2006, Part III, LNAI 4253, 2006, pp. 1280–1287.

[11] K. Langendoen, N. Reijers, Distributed localization in wireless sensor networks: a quantitative comparison, Computer Networks 43 (4) (2003) 499–518.

[12] J. Lee, W. Chung, E. Kim, Structure learning of bayesian networks using dual genetic algorithm, IEICE Transactions on Information and System E91-D (1) (2007) 32–43.

[13] J. Lee, B. Choi, E. Kim, A novel range-free localization based on multi-dimensional support vector regression trained in the primal space, IEEE Transactions on Neural Networks and Learning Systems 24 (7) (2013) 1–15.

[14] J. Li, Q.-S. Jia, X. Guan, X. Chen, Tracking a moving object via a sensor network with a partial information broadcasting scheme, Information Sciences 181 (20) (2011) 4733–4753.

[15] H. Lim, J. Hou, Localization for anisotropic sensor networks, in: Proceedings of IEEE INFOCOM '05, vol. 1, 2005, pp. 138–149.

[16] Y. Liu, X. Yi, Y. He, A novel centroid localization for wireless sensor networks, International Journal of Distributed Sensor Networks 2012 (829253) (2012) 1–8.

[17] G. Mao, B. Fidan, B.D.O. Anderson, Wireless sensor network localization techniques, Computer Networks 51 (10) (2007) 2529–2553.

[18] D. Musicant, A. Feinberg, Active set support vector regression, IEEE Transactions on Neural Networks 15 (2) (2004) 268–275.

[19] R. Nagpal, H. Shrobe, J. Bachrach, Organizing a global coordinate system from local information on an ad hoc sensor network, in: Proceedings of Information Processing in Sensor Networks, 2003, pp. 333–348.

[20] X. Nguyen, M.I. Jordan, B. Sinopoli, A kernel-based learning approach to ad hoc sensor network localization, ACM Transaction on Sensor Networks 1 (1) (2005) 134–152.

[21] D. Niculescu, B. Nath, DV based positioning in ad hoc networks, Journal of Telecommunication Systems 22 (2003) 267–280.

[22] J.J. Pan, J.T. Kwok, Y. Chen, Multidimensional vector regression for accurate and low-cost location estimation in pervasive computing, IEEE Transactions on Knowledge and Data Engineering 18 (9) (2006) 1181–1193.

[23] N. Pogkas, G. Karastergios, C. Antonopoulos, S. Koubias, G. Papadopoulos, An ad-hoc sensor network for disaster relief operations, in: Proceedings of IEEE Conference on Emerging Technologies and Factory Automation, vol. 2, 2005, pp. 131–139.

[24] P.K. Sahoo, J.-P. Sheu, K.-Y. Hsieh, Target tracking and boundary node selection algorithms of wireless sensor networks for internet services, Information Sciences 230 (2013) 21–38.

[25] M. Sánchez-Fernández, M. de-Prado-Cumplido, J. Arenas-Garcıa, F. Pérez-Cruz, SVM multiregression for nonlinear channel estimation in multiple-input multiple-output systems, IEEE Transactions on Signal Processing 52 (8) (2004) 2298–2307.

[26] A. Savvides, C. Han, M.B. Strivastava, Dynamic fine grained localization in ad-hoc networks of sensors, in: Proceedings of the International Conference on Mobile Computing and Networking, 2001, pp. 166–179.

[27] B. Schölkopf, A. Smola, Learning with Kernels, MIT Press, Cambridge, MA, 2001.

[28] Y. Shang, W. Ruml, Y. Zhang, M. Fromherz, Localization from connectivity in sensor networks, IEEE Transactions on Parallel and Distributed Systems 15 (11) (2004) 961–974.

[29] J.P. Sheu, P.C. Chen, C.S. Hsu, A distributed localization scheme for wireless sensor networks with improved grid-scan and vector-based refinement, IEEE Transactions on Mobile Computing 7 (9) (2008) 1110–1123.

[30] A. Shilton, B. Sundaram, M. Palaniswami, Ad-hoc wireless sensor network localization using support vector regression, in: Proceedings of ICT Mobile Summit, Stockholm, June, 2008.

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

[32] D.A. Tran, T. Nguyen, Localization in wireless sensor networks based on support vector machines, IEEE Transactions on Parallel and Distributed Systems 19 (7) (2008) 981–994.

[33] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, NY, 1995.

[34] H. Wu, J. Chen, C. Wang, Y. Sun, A kernel-based localization approach in wireless sensor networks, in: Proceedings of 2008 Second International Conference on Future Generation Communication and Networking, 2008, pp. 31–34.

[35] K. Yedavalli, B. Krishnamachari, Sequence-based localization in wireless sensor networks, IEEE Transactions on Mobile Computing 7 (1) (2008) 81–94.

[36] S. Yun, J. Lee, W. Chung, E. Kim, S. Kim, A soft computing approach to localization in wireless sensor networks, Expert Systems with Applications 36 (4) (2009) 7552–7561.

[37] Z. Zhong, T. He, RSD: A metric for achieving range-free localization beyond connectivity, IEEE Transactions on Parallel and Distributed Systems 22 (11) (2011) 1943–1951.