Multidimensional Vector Regression for Accurate and Low-Cost Location Estimation in Pervasive Computing

Jeffrey Junfeng Pan, James T. Kwok, Qiang Yang, *Senior Member*, *IEEE*, and Yiqiang Chen

Abstract—In this paper, we present an algorithm for multidimensional vector regression on data that are highly uncertain and nonlinear, and then apply it to the problem of indoor location estimation in a wireless local area network (WLAN). Our aim is to obtain an accurate mapping between the signal space and the physical space without requiring too much human calibration effort. This location estimation problem has traditionally been tackled through probabilistic models trained on manually labeled data, which are expensive to obtain. In contrast, our algorithm adopts Kernel Canonical Correlation Analysis (KCCA) to build a nonlinear mapping between the signal-vector space and the physical location space by transforming data in both spaces into their canonical features. This allows the pairwise similarity of samples in both spaces to be maximally correlated using kernels. We use a Gaussian kernel to adapt to the noisy characteristics of signal strengths and a Matérn kernel to sense the changes in physical locations. By using real data collected in an 802.11 wireless LAN environment, we achieve accurate location estimation for pervasive computing while requiring a much smaller set of labeled training data than previous methods.

Index Terms—Location-dependent and sensitive, correlation and regression analysis, pervasive computing.

1 Introduction

In this paper, we present an algorithm for multidimensional vector regression on data that are highly uncertain and nonlinear, and apply the algorithm to the problem of indoor-location estimation in a 802.11 wireless network. The problem of indoor location estimation can be considered as one of building a mapping from radio-frequency signals to a multivalued function that corresponds to locations. Indoor location estimation is a major area of pervasive computing applications that range from context-dependent content delivery to object tracking [1] and people monitoring [2], [3]. Many systems utilize the signal strength values received from the access points to infer the location of mobile devices [4], [5], [6], [7], [8], [9]. Similarly, in a sensor network, the location information must also be inferred from signals received from various deployed sensors [10], [11].

In general, location-estimation systems using radio frequency (RF)-based signal-strength values function in two phases: an *offline training phase* and an *online localization phase*. In the offline training phase, data are collected that correspond to each location and labeled by hand with the location information. Then, a model is trained by considering the signal strength values received from the access points at selected locations in the area of interest. In the online localization phase, the learned model is applied to new signals. The real-time signal strength samples received from the access points are used to estimate the current location based on the learned model.

Models used in location-estimation systems could be broadly categorized into two classes: *Radio-Propagation Models* and *Empirical-Fit Models*. For building *Radio-Propagation Models*, we usually try to identify different factors that may affect the fluctuation and attenuation of radio signal and encode these factors as parameters into radio propagation equations [12], [13], [14], [15]. For building *Empirical-Fit Models*, we would empirically describe the signals in terms of their mean values, histogram, or spline interpolation without formulating a closed-form propagation model [4], [16], [6], [7], [8]. In order to calibrate both kinds of models, efforts need be spent to collect signal samples from different locations. Thus, how to maintain a high-level of accuracy while reducing calibration effort is a challenging task, since the data collection process is time-consuming. For example, in our earlier test, it took many hours to collect and label the signal-strength data in a small indoor environment. Furthermore, the mapping between the signal and physical location spaces is very difficult to construct with only limited labeled data due to uncertainty and nonlinearity. Nonlinearity exists when similar locations have very different signal signatures. Therefore, a direct mapping between the two spaces may not always work well, even when much data are collected.

In this paper, we present a multidimensional vector regression method for building a mapping function by addressing the problems of uncertainty and nonlinearity directly. Our main intuition is to perform a kernel-based transformation of the signal and physical location spaces to capture the nonlinear relationship between the signals and

- *J.J. Pan, J.T. Kwok, and Q. Yang are with the Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong SAR. E-mail: {panjf, jamesk, qyang}@cs.ust.hk.*
- *Y. Chen is with the Institute of Computing Technology, Shanghai Branch, Chinese Academy of Sciences, ZhangJiang Science Park, Shanghai, China. E-mail: yqchen@ict.ac.cn.*

Manuscript received 5 July 2005; revised 26 Jan. 2006; accepted 4 Apr. 2006; published online 19 July 2006.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0259-0705.

1041-4347/06/$20.00 © 2006 IEEE Published by the IEEE Computer Society

locations. Furthermore, we perform kernel canonical correlation analysis (KCCA) in the two spaces for feature extraction. This allows the pairwise similarity of samples in both spaces to be maximally correlated. We use a Gaussian kernel to adapt to the noisy characteristics of signal strengths and a Matérn kernel to sense the changes in physical locations. In comparison to previous methods, a major advantage of our proposed technique is that we can obtain higher accuracy while reducing the training cost by requiring only a fraction of the labeled samples.

Although this paper focuses on the location estimation problem in pervasive computing, the proposed method has more general practical implications. In effect, we are addressing a new type of machine learning problem where there are many classes but a limited amount of training examples. For this type of problem, traditional machine learning algorithms will usually result in overfitting. Our idea is to make use of the inherent relationships between the different classes. In this paper, we rely on the similarity relationship imposed by distance relations among the different locations. As will be shown later in the paper, this relationship can be captured using kernel functions [17]. Consequently, the accuracy of the classification model can be greatly enhanced with the use of this new piece of information.

The rest of this paper is organized as follows: Section 2 first reviews the works on location estimation, multidimensional vector regression, and (kernel) canonical correlation analysis. Section 3 then describes the proposed KCCA-based location estimation algorithm. Results on a series of WLAN location estimation experiments using data collected in a realistic environment are presented in Section 4, and the last section gives some concluding remarks. A preliminary version of this paper has appeared in [18].

2 Related Works

2.1 Location Estimation

Location estimation systems based on radio techniques could be classified into different categories according to different criteria. Considering the system architecture, they could be client-based [4], [6], [14], [8] or infrastructure-based [19], [16]. When emphasizing the use of probability, they could be deterministic [4], [20], [7], [21] or distribution-based [22], [13], [6], [23], [15], [24]. In this paper, we follow [4] and classify location estimation systems into two main categories: (1) *Radio-Propagation Models* and (2) *Empirical-Fit Models*. The latter may not rely on the knowledge of radio propagation.

Location estimation systems that adopt *Radio-Propagation Models* benefit from the knowledge of radio propagation. Observing the nonlinear and noisy patterns of radio signal strength, people who develop these kind of systems usually go to a *low-level* analysis and try to identify the hidden factors that cause these patterns. Typical factors include path loss, shadowing, and multipath, which lead to log-distance, log-normal, and more sophisticated multipath channel models [25], [26], [13], [23]. Generally speaking, a more accurate formula needs additional information about the physical environment and the network. For example, the location of access points needs to be given in many radio propagation models. When the building structure (or even the material) is known, the wall attenuation factors and the corridor effect could be encoded and form a more accurate propagation model [4], [14]. Collecting environment information may raise the calibration effort. There are many algorithms that apply radio propagation models. Take trilateration, for example; we could first transform signal into distance information from a client to a small number of access points (ranging). Then, a least square fit is used to estimate the most probable coordinate [11]. As an alternative, RADAR [4] discretizes the localization area to grid locations and *theoretically* computes the signal strength at these locations in the offline phase. Then, nearest-neighbor heuristics and triangulation methods are used to infer a client's location in the online phase. Maligan et al. [14] explore that different access points should behave similarly. Their work presents a Bayesian network model that encodes knowledge about radio-propagation models, which make use of similarity among the access points and other factors. It also points out that, by incorporating additional knowledge such as the motion constraint of a user, the calibration effort can be further reduced. Although we are not focused on identifying all these factors and encoding them into models in this paper, we take a complementary view by discovering the correlation between signal and location spaces and modeling this correlation with multi-dimensional vector regression.

Another class of location estimation systems is based on *Empirical-Fit Models*, which employ machine learning techniques. Path loss, shadowing, and multipath are caused by a complicated underlying process in a complex environment. When all these factors are mixed together, they show a *high-level* of nonlinear and noisy patterns. These patterns can be captured when sufficient empirical data are manually collected at different locations [8], [22] or automatically by additional hardware [16], [7], [27]. These methods need less information about physical layout and network configuration. The input is usually implicitly encoded into radio maps [8] at different locations. In such cases, the locations of access points are not needed. Typical pattern descriptions include histogram [8], [15], [22], mixture of Gaussian [15], kernel matrix [18], Akima Spline [16], or simply the mean value of signal strength at different locations [4], [7]. For example, the LANDMARC system [7] uses reference tags to dynamically construct and update radio map. It alleviates the effects caused by the fluctuation in RF signal strength. The method first computes the distances between the signal-strength vectors received from the tracking tags and those from different reference tags, respectively. It then uses k nearest reference tags' coordinates to calculate the approximate coordinate of the tracking tag. A similar technique is used in LEASE [16], [27].

More recently, intense research efforts have been taken to seek additional knowledge in order to boost the accuracy of location determination systems. For example, [28], [10], [6], [14], [24] take the sequential characteristics of user traces into consideration by posing a reasonable assumption that a user could not walk irregularly, run too fast, or go through a wall. These models come closer to the general framework of

TABLE 1
A Summary of the Characteristics of Some Common Location Estimation Methods

Method	Propagation-based	Signal dynamics	User dynamics
Multilateration [11], Statistical [23], Fingerprinting [13], RADAR [4]	✓		
RADAR+ [28], Bayesian-Indoor [14]	✓	✓	
Active-Campus [21], PHD [20], Robust-Indoor [12], RADAR+ [28]	✓		✓
RADAR [4], Horus [8], Probabilistic [15], KCCA [18]			
LEASE[16], Reduce[22], Horus+[30], Adaptive [27], LANDMARC [7]		✓	
RADAR+ [28], MCL [10], Robotics [6], State-Machine [24]			✓

Bayesian filtering [29]. For example, Ladd et al. [6] suggest a sensor fusion model and show a strong correlation between consecutive locations. The robotics-based location sensing system in [6] applies Bayesian inference to compute the conditional probabilities over locations based on received signal-strength samples from various access points. Then, a postprocessing step, which utilizes the spatial constraints of a user's movement trajectories, refines the location estimation and rejects the estimates that show significant changes in the physical location space. Depending on whether the postprocessing step is used or not, the accuracy of this method is 83 percent or 77 percent within 1.5 meters. Bahl et al. [28], [22], [16], [7], [27], [30] study the dynamic features of signal strength and update their models to adapt the change. Krishnan et al. [16], [7], [27] employ additional hardware such as sniffers to help recalibrate the radio map periodically. Bahl et al. [28] improve the localization performance by profiling the radio characteristics into "busy" and "nonbusy" hours. Chai and Yang [22], [14] could use unlabeled data to advance the performance so that they are capable of dynamically updating their model to fit the characteristics of the radio environment. Youssef and Agrawala [30] use an autoregressive model to reduce fluctuation between consecutive signal strength.

In practice, location-estimation systems may combine both Radio-Propagation and Empirical-Fit models, and capture the signals and the user dynamics by filtering in both signal and location spaces. Thus, the above models can *complement* each other and improve the performance. A summary of related works is shown in Table 1. The table shows a high-level overview of different location-estimation algorithms. Thus, items in the table only highlight the main and common features of these algorithms. In this table, some papers propose more than one algorithm so that they fall into multiple categories. For example, RADAR [4] and its enhanced version [28] have two variants: One is based on Radio Propagation and the other is based on Empirical Fit, although both use K-Nearest-Neighbor in the online localization phase.

2.2 Multidimensional Vector Regression

There are a variety of application problems where multiple outputs are to be predicted using multidimensional inputs. They form a *multidimensional vector regression* or *vector-valued function* learning problem [31], [32]. Take location estimation, for example; a user's position, which is represented by a two-dimensional vector (x, y), can be inferred using multiple signal sources from nearby access points by trilateration: first, a nonlinear transformation from signal strength to distance and, second, a least square method to recover the coordinate. Dependency between and within all the components of signal and location vectors may be highly nonlinear, partially depending on the spatial structure of the localization area of interest. For example, in Fig. 2a, there are four hallways forming an irregular shape. For a certain predicted value of x, some y values should automatically be ruled out (and vice versa) because the user can only be physically located at a certain hallway. Similar situations also exist in many real-world domains, such as medical care, manufacturing, or stock prediction, where multiple correlated inputs and outputs are involved [33]. Thus, a key question is how to improve prediction accuracy by taking advantage of the possible correlations between components of an output vector.

Breiman and Friedman [33] introduced the *curds and whey* method, which bridges traditional scalar-valued regression and vector-valued regression by a two-stage procedure. First, each output dimension is separately estimated by least squares fitting or ridge regression. Second, all output dimensions are then combined together to exploit possible correlations among the output dimensions. Breiman and Friedman [33] show that the prediction error can be substantially reduced when there are correlations between outputs. Even if the outputs are indeed uncorrelated, the *curds and whey* method does not lead to performance degradation. Furthermore, once we find that the accuracy is not improved due to weak or no correlation among outputs, the second step can be disabled and a traditional regression method can be used instead. The *curds and whey* method can be generalized for nonlinear regression [34].

Partial least squares (PLS) [35] is another vector regression method popularly used in chemometrics. It is based on maximizing the covariance between the (multidimensional) inputs and outputs. It generalizes and combines features from principal component analysis and multiple linear regression. The data are first projected onto the principal directions, which is then followed by least squares fitting in this projected space. When all principal directions are used, PLS degenerates to standard multiple linear regression. Compared to the ordinary least squares method, PLS prevents overfitting by properly choosing the number of principal directions. Recently, a nonlinear extension of PLS has also been proposed by exploiting the kernel trick [36].

Another well-known vector regression method is *canonical correlation analysis* (CCA) [37], [32]. It tries to find a pair of *canonical vectors*, one for the input and another for the output, so that the projections onto these canonical vectors are maximally correlated. Compared to PLS, CCA exploits the correlation, rather than the covariance, within and between the inputs and outputs. In many real-world applications, the inputs often come from multiple and heterogenous sensors, which have different scales and/or noise levels. A large covariance between signals may not necessarily mean a strong correlation. It is possible that a pair of principal directions in the two spaces have high covariance merely because the signal range is large and/or the noise level is high. Conversely, a pair of directions may have perfect correlation but a low covariance. In such cases, *correlation* (and, thus, CCA), rather than *covariance* (PLS), should be used. While CCA considers only linear correlation, its kernelized version (*kernel CCA*) [38] can also exploit nonlinear correlation. Both CCA and kernel CCA will be discussed in more detail in the next section.

2.3 (Kernel) Canonical Correlation Analysis

2.3.1 Canonical Correlation Analysis (CCA)

Given two sets of variables $\mathbf{x}$ and $\mathbf{y}$, canonical correlation analysis (CCA) [37] attempts to find a basis for each set such that the correlation between the projections of the variables onto these basis vectors are mutually maximized. Mathematically, given n instances

$$S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\},$$

CCA finds directions (*canonical vectors*) $\mathbf{w}_\mathbf{x}$ and $\mathbf{w}_\mathbf{y}$ so that the sets of transformed variables (*canonical variates*)

$$S_\mathbf{x} \equiv S_\mathbf{x}(\mathbf{w}_\mathbf{x}) = (\langle \mathbf{w}_\mathbf{x}, \mathbf{x}_1 \rangle, \langle \mathbf{w}_\mathbf{x}, \mathbf{x}_2 \rangle, \ldots, \langle \mathbf{w}_\mathbf{x}, \mathbf{x}_n \rangle)$$

and

$$S_\mathbf{y} \equiv S_\mathbf{y}(\mathbf{w}_\mathbf{y}) = (\langle \mathbf{w}_\mathbf{y}, \mathbf{y}_1 \rangle, \langle \mathbf{w}_\mathbf{y}, \mathbf{y}_2 \rangle, \ldots, \langle \mathbf{w}_\mathbf{y}, \mathbf{y}_n \rangle)$$

are maximally correlated.

Define the total covariance matrix by[1]

$$\mathbf{C} = \hat{\mathbb{E}}\left[\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}'\right] = \begin{bmatrix}\mathbf{C}_{\mathbf{xx}} & \mathbf{C}_{\mathbf{xy}}\\ \mathbf{C}_{\mathbf{yx}} & \mathbf{C}_{\mathbf{yy}}\end{bmatrix},$$

where $\hat{\mathbb{E}}[\cdot]$ is the empirical expectation operator, $\mathbf{C}_{\mathbf{xx}}, \mathbf{C}_{\mathbf{yy}}$ are the within-sets covariance matrices, and $\mathbf{C}_{\mathbf{xy}} = \mathbf{C}'_{\mathbf{yx}}$ is the between-sets covariance matrix. The correlation coefficient between $S_\mathbf{x}$ and $S_\mathbf{y}$ can be written as

$$\rho = \frac{\hat{\mathbb{E}}[\langle \mathbf{w}_\mathbf{x}, \mathbf{x}\rangle\langle \mathbf{w}_\mathbf{y}, \mathbf{y}\rangle]}{\sqrt{\hat{\mathbb{E}}[\langle \mathbf{w}_\mathbf{x}, \mathbf{x}\rangle^2]\hat{\mathbb{E}}[\langle \mathbf{w}_\mathbf{y}, \mathbf{y}\rangle^2]}} = \frac{\mathbf{w}'_\mathbf{x}\hat{\mathbb{E}}[\mathbf{x}\mathbf{y}']\mathbf{w}_\mathbf{y}}{\sqrt{\mathbf{w}'_\mathbf{x}\hat{\mathbb{E}}[\mathbf{x}\mathbf{x}']\mathbf{w}_\mathbf{x} \cdot \mathbf{w}'_\mathbf{y}\hat{\mathbb{E}}[\mathbf{y}\mathbf{y}']\mathbf{w}_\mathbf{y}}} \quad (1)$$

$$= \frac{\mathbf{w}'_\mathbf{x}\mathbf{C}_{\mathbf{xy}}\mathbf{w}_\mathbf{x}}{\sqrt{\mathbf{w}'_\mathbf{x}\mathbf{C}_{\mathbf{xx}}\mathbf{w}_\mathbf{x} \cdot \mathbf{w}'_\mathbf{y}\mathbf{C}_{\mathbf{yy}}\mathbf{w}_\mathbf{y}}}. \quad (2)$$

It can be shown that $\mathbf{w}_\mathbf{x}$ can be obtained by solving the generalized eigenproblem [38]

$$\mathbf{C}_{\mathbf{xy}}\mathbf{C}_{\mathbf{yy}}^{-1}\mathbf{C}_{\mathbf{yx}}\mathbf{w}_\mathbf{x} = \lambda^2\mathbf{C}_{\mathbf{xx}}\mathbf{w}_\mathbf{x}.$$

Subsequently, $\mathbf{w}_\mathbf{y}$ can then be obtained as $\mathbf{w}_\mathbf{y} = \frac{1}{\lambda}\mathbf{C}_{\mathbf{yy}}^{-1}\mathbf{C}_{\mathbf{yx}}\mathbf{w}_\mathbf{x}$. Moreover, it can be shown that the λ obtained is equal to the ρ in (1).

2.3.2 Kernel Canonical Correlation Analysis (KCCA)

A major limitation of CCA is that it can only exploit linear relationships between $\mathbf{x}$ and $\mathbf{y}$. As is now well-known, the use of kernels offers efficient, nonlinear extensions for many standard linear procedures [17]. In kernel canonical correlation analysis (KCCA), [38] implicitly maps $\mathbf{x}$ and $\mathbf{y}$ to $\phi_x(\mathbf{x})$ and $\phi_y(\mathbf{y})$, and then performs traditional CCA in the two high-dimensional feature spaces. Using the dual representations for the projection directions $\mathbf{w}_{\phi_x(\mathbf{x})}$ and $\mathbf{w}_{\phi_y(\mathbf{y})}$:

$$\mathbf{w}_{\phi_x(\mathbf{x})} = \mathbf{S}'_{\phi_x(\mathbf{x})}\alpha, \quad \text{and} \quad \mathbf{w}_{\phi_y(\mathbf{y})} = \mathbf{S}'_{\phi_y(\mathbf{y})}\boldsymbol{\beta},$$

where

$$\mathbf{S}_{\phi_x(\mathbf{x})} = [\phi_x(\mathbf{x}_1), \ldots, \phi_x(\mathbf{x}_n)]', \ \mathbf{S}_{\phi_y(\mathbf{y})} = [\phi_y(\mathbf{y}_1), \ldots, \phi_y(\mathbf{y}_n)]',$$

and $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{R}^n$. Denote the corresponding kernel functions by $k_\mathbf{x}(\cdot,\cdot)$ and $k_\mathbf{y}(\cdot,\cdot)$, and the kernel matrices (defined on all n instances) by $\mathbf{K}_\mathbf{x}$ and $\mathbf{K}_\mathbf{y}$. The kernelized counterpart of (2) is

$$\rho = \frac{\boldsymbol{\alpha}'\mathbf{K}_\mathbf{x}\mathbf{K}_\mathbf{y}\boldsymbol{\beta}}{\sqrt{\boldsymbol{\alpha}'\mathbf{K}_\mathbf{x}^2\boldsymbol{\alpha} \cdot \boldsymbol{\beta}'\mathbf{K}_\mathbf{y}^2\boldsymbol{\beta}}}, \quad (3)$$

which is then maximized with regard to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. However, it can be shown that one can always obtain perfect correlation and, consequently, an uninteresting result given that the kernel matrices $\mathbf{K}_\mathbf{x}$ and $\mathbf{K}_\mathbf{y}$ are invertible [38]. To control the flexibility of the projections, the norms of the associated weight vectors are thus penalized as in other kernel methods. Hence, instead of maximizing (3), we maximize the regularized version

$$\frac{\boldsymbol{\alpha}'\mathbf{K}_\mathbf{x}\mathbf{K}_\mathbf{y}\boldsymbol{\beta}}{\sqrt{(\boldsymbol{\alpha}'\mathbf{K}_\mathbf{x}^2\boldsymbol{\alpha} + \kappa\boldsymbol{\alpha}'\mathbf{K}_\mathbf{x}\boldsymbol{\alpha}) \cdot (\boldsymbol{\beta}'\mathbf{K}_\mathbf{y}^2\boldsymbol{\beta} + \kappa\boldsymbol{\beta}'\mathbf{K}_\mathbf{y}\boldsymbol{\beta})}}, \quad (4)$$

where κ is a user-defined regularization parameter. It can be shown that $\boldsymbol{\alpha}$ can be obtained by solving the generalized eigenproblem

$$(\mathbf{K}_\mathbf{x} + \kappa\mathbf{I})^{-1}\mathbf{K}_\mathbf{y}(\mathbf{K}_\mathbf{y} + \kappa\mathbf{I})^{-1}\mathbf{K}_\mathbf{x}\boldsymbol{\alpha} = \lambda^2\boldsymbol{\alpha}, \quad (5)$$

1. In this paper, vector/matrix transpose is denoted by the superscript $'$.

where $\mathbf{I}$ is the identity matrix. Subsequently, β can be obtained as

$$\beta = \frac{1}{\lambda}(\mathbf{K_y} + \kappa\mathbf{I})^{-1}\mathbf{K_x}\boldsymbol{\alpha}. \quad (6)$$

Given any $\tilde{\mathbf{x}}$, its projection on $\mathbf{w}_{\phi_x(\mathbf{x})}$ is given by

$$P_{\mathbf{x}}(\tilde{\mathbf{x}}) = \phi_x(\tilde{\mathbf{x}})'\mathbf{w}_{\phi_x(\mathbf{x})} = \mathbf{k}_{\tilde{\mathbf{x}}'}\boldsymbol{\alpha}, \quad (7)$$

where $\mathbf{k}_{\tilde{\mathbf{x}}} = [k_{\mathbf{x}}(\tilde{\mathbf{x}}, \mathbf{x}_1), k_{\mathbf{x}}(\tilde{\mathbf{x}}, \mathbf{x}_2), \ldots, k_{\mathbf{x}}(\tilde{\mathbf{x}}, \mathbf{x}_n)]'$. Similarly, the projection of any $\tilde{\mathbf{y}}$ onto $\mathbf{w}_{\phi_y(\mathbf{y})}$ is

$$P_{\mathbf{y}}(\tilde{\mathbf{y}}) = \phi_y(\tilde{\mathbf{y}})'\mathbf{w}_{\phi_y(\mathbf{y})} = \mathbf{k}_{\tilde{\mathbf{y}}'}\boldsymbol{\beta},$$

where $\mathbf{k}_{\tilde{\mathbf{y}}} = [k_{\mathbf{y}}(\tilde{\mathbf{y}}, \mathbf{y}_1), k_{\mathbf{y}}(\tilde{\mathbf{y}}, \mathbf{y}_2), \ldots, k_{\mathbf{y}}(\tilde{\mathbf{y}}, \mathbf{y}_n)]'$.

The generalized eigenproblem in (5) can be solved by using the (complete) Cholesky decomposition [39]. However, the kernel matrices $\mathbf{K_x}$ and $\mathbf{K_y}$ are of size n and, so, obtaining the Cholesky decomposition can become computational expensive for large training sets. In this case, the incomplete Cholesky decomposition or partial Gram-Schmidt orthogonalization (PGSO) can be used instead [38]. The basic idea is to find a low-rank approximation of the kernel matrix, and the incomplete Cholesky decomposition differs from the complete Cholesky decomposition in that all pivots below a certain threshold are simply skipped. The decomposition is obtained by picking columns of the kernel matrix one at a time, at each step choosing the column that leads to the greatest reduction in the approximation error. Once a column (or basis vector) is selected, the other columns are then orthogonalized by the Gram-Schmidt algorithm.

3 Location Estimation Using KCCA

3.1 Motivation

Consider the two-dimensional location estimation problem.[2] Its goal is to obtain a mapping between the space of signal strengths obtained at p access points

$$\mathcal{S} = \{\mathbf{s} \equiv [s_1, s_2, \ldots, s_p]' \in \mathbb{R}^p\} \quad (8)$$

and the physical location space $\mathcal{P} = \{\ell \equiv [x, y]' \in \mathbb{R}^2\}$. Methods such as trilateration deal with the mapping between signal and location spaces in two steps: First, transform signal into distance with a nonlinear equation (ranging) and, second, recover the most probable coordinate from distance with least square methods. In this paper, we directly build a nonlinear mapping between signal and location spaces. We consider x and y together and emphasize the *correlation* between signal and physical location spaces, observing that the pairwise similarity in the signal space should match the pairwise similarity in the physical location space. For example, in Fig. 1, signal S_A should be more similar to S_B than S_C, since A is closer to B in the physical location space. Consequently, we consider both x and y together and use KCCA to learn the mapping between the two spaces.

2. Extension to the three-dimensional (or even higher-dimensional) case is straight-forward.

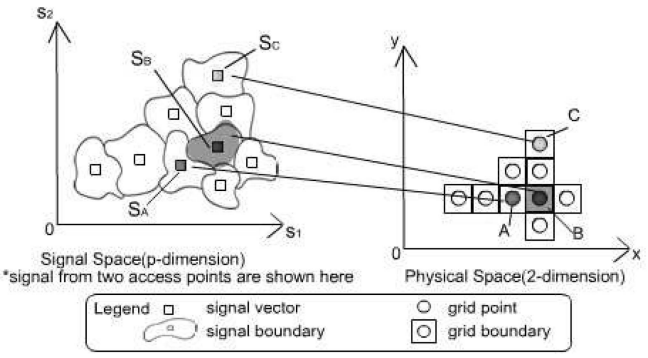


Fig. 1. Correlation between the signal and physical location spaces.

3.2 The LE-KCCA Algorithm

In contrast to the previous works described in Section 2.1, our approach builds a similarity-based mapping function by making full use of the continuous location information in kernel-based transformation.

3.2.1 Offline Training Phase

In the training phase, the following steps are taken:

1. Signal strengths are collected at various grid locations.
2. KCCA, with appropriate choices for the two kernels (Section 3.3), is used to learn the relationship between the signal and physical location spaces. In particular, λ_is and $\boldsymbol{\alpha}_i$s are obtained from the generalized eigenproblem in (5), and the corresponding $\boldsymbol{\beta}_i$'s from (6).
3. For each training pair $(\mathbf{s}_i, \boldsymbol{\ell}_i)$, its projections

$$P(\mathbf{s}_i) = [P_1(\mathbf{s}_i), P_2(\mathbf{s}_i), \ldots, P_T(\mathbf{s}_i)]' \quad (9)$$

on the T canonical vectors are obtained from (7).

3.2.2 Online Localization Phase

In the localization phase, the location of a new signal strength vector $\tilde{\mathbf{s}}$ is estimated as follows:

1. Use (7) to project $\tilde{\mathbf{s}}$ onto the canonical vectors and obtain

$$P(\tilde{\mathbf{s}}) = [P_1(\tilde{\mathbf{s}}), P_2(\tilde{\mathbf{s}}), \ldots, P_T(\tilde{\mathbf{s}})]'.$$

2. Among the projections (9) from the training samples, find the K neighbors closest to $P(\tilde{\mathbf{s}})$. In this paper, the weighted Euclidean distance

$$d_i = \sum_{j=1}^{T} \lambda_j (P_j(\tilde{\mathbf{s}}) - P_j(\mathbf{s}_i))^2 \quad (10)$$

is employed in finding the neighbors.
3. Interpolate these neighbors' physical locations to predict the physical location of $\tilde{\mathbf{s}}$. In this paper, we simply output the median (or mean for continuous location estimation) of the (x, y) coordinates of these K neighbors.

Note that this is essentially a variant of the weighted K-nearest neighbor method. Here, we use λ_j as the weight in (10). As mentioned in Section 2.3.1, λ_j is equal to the correlation coefficient ρ in (1), and thus serves as a

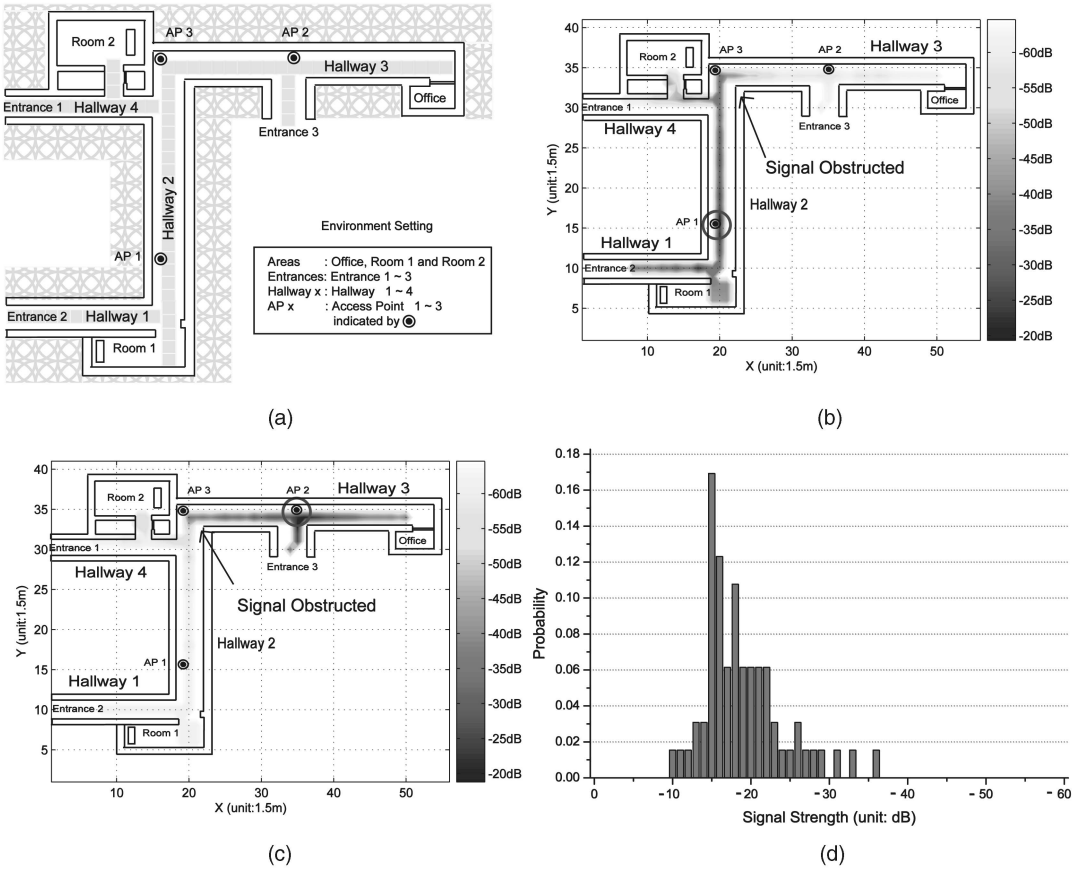


Fig. 2. Experimental test-bed and radio propagation characteristics. (a) Layout of the experimental test-bed. (b) Signal distribution from AP 1. (c) Signal distribution from AP 2. (d) Signal distribution at a fixed location.

reasonable measure for the importance of the corresponding canonical vector.

3.3 Choice of Kernels

The choice of kernels is highly dependent on the nonlinear and noisy characteristics of the location estimation problem we are addressing due to possible path loss, shadowing, and multipath effects, etc. [25]. In Fig. 2a, there are three access points (APs). Figs. 2b and 2c show the average signal strength distributions of AP1 and AP2, respectively. As can be seen, the signal strength changes sharply (nonlinear) at the corner of intersecting hallways 2 and 3 due to the shadowing effect of the walls. Fig. 2d shows the typical signal distribution at a particular location from a fixed access point. As can be seen, this noisy signal can be as weak as -6dB and as strong as -10dB. Its empirical distribution (*radio map*) is thus difficult to obtain, especially when the training samples are scarce. As a first approximation, the Gaussian distribution has been used in characterizing the nonlinearity of the signal strengths [15]. Hence, in this paper, we also use the Gaussian kernel

$$G(\mathbf{s}_1, \mathbf{s}_2) = \exp(-w_G^2 \|\mathbf{s}_1 - \mathbf{s}_2\|^2) \tag{11}$$

for the signal space. Here, $\|\cdot\|$ denotes the Euclidean norm and w_G is a user-defined parameter that reflects the smoothness of the radio map.

On the other hand, measurements of the physical locations are relatively clean. Intuitively, it seems that Euclidean distance best represents the "similarity" between two locations. Unfortunately, the Euclidean function is not a valid kernel since a valid kernel should satisfy the positive definite property [40]. For the commonly used Gaussian kernel, it has been argued by [41] that sample paths of the Gaussian model are "infinitely smooth," thus often leading to unreasonably low predictive variance [42]. Consider the Euclidean distance $|x_i - x_j|$. This distance measure cannot be used directly as a kernel without first transforming it into a valid kernel. Therefore, in this paper, we adopt the Matérn kernel, which is a function that reflects the Euclidean distance (Fig. 3) [42].

$$M(\mathbf{x}_1, \mathbf{x}_2) = \frac{2(\sqrt{\nu} w_M \|\mathbf{x}_1 - \mathbf{x}_2\|)^\nu}{\Gamma(\nu)} K_\nu\big(2\sqrt{\nu} w_M \|\mathbf{x}_1 - \mathbf{x}_2\|\big). \tag{12}$$

Here, ν is a user-defined smoothness parameter, $\Gamma(\nu)$ is the Gamma function $\Gamma(\nu) = \int_0^\infty e^{-t} t^{\nu-1} dt$, and $K_\nu(r)$ is the modified Bessel function of the second kind with degree ν [39]:

$$K_{\nu(r)} = \Big(\frac{r}{2}\Big)^\nu \sum_{k=0}^{\infty} \frac{(\frac{r^2}{4})^k}{k!\Gamma(\nu + k + 1)}.$$

It can be shown that, when $\nu \to \infty$, the Matérn kernel degenerates to the Gaussian kernel. On the other hand, when $\nu = 0.5$, it degenerates to the exponential kernel

$$E(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\sqrt{2} w_M \|\mathbf{x}_1 - \mathbf{x}_2\|).$$

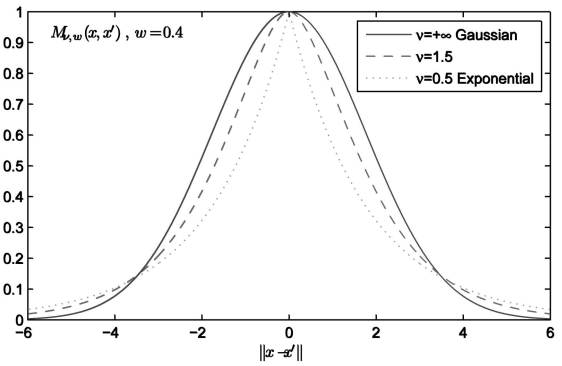


Fig. 3. The Matérn kernel.

Note that both the Gaussian and Matérn kernels are isotropic[3] and, so, are invariant to the location of origin and to arbitrary rotation.

3.4 Time Complexity Analysis

In location estimation, the training phase can be performed offline and, so, its speed is not very important. On the other hand, the localization phase has to be performed online. In this section, we compare the time complexities for online localization as required by LE-KCCA and other popular location estimation algorithms, namely, support vector machine (SVM) [43], support vector regression (SVR) [43], model trees [44], maximum likelihood estimation (MLE) [8], and RADAR [4]. These algorithms will also be compared experimentally in Section 4.

In the following, let A be the number of access points, L the number of locations, V the number of support vectors, S the number of training samples, and T the number of canonical vectors.

1. LE-KCCA: Localization involves three steps (Section 3.2.2). The first step projects $\tilde{\mathbf{s}}$ onto the T canonical vectors, each being a linear combination of the S training samples in the feature space.[4] The time for one kernel evaluation is $O(A)$. Thus, this step takes $O(AST)$ time. In the second step, we compute the weighted Euclidean distance, and this takes $O(ST)$ time. These distances are then ranked, but as ranking mainly involves comparison, its time is small compared to those of the others and so will be dropped. The total time complexity is thus $O(AST + ST)$.
2. SVM: Our SVM treats location estimation as a multiple-class classification problem, by using the signal strengths from the A access points as input and the L locations as output. There are $O(V)$ support vectors at each location.
3. SVR: Again using the signal strengths from the A access points as input, our SVR builds a regression model for each output dimension, x and y. As there are only two outputs, the time complexity for localization is $O(AV)$.
4. Model tree: We build two model trees, one for each output dimension. The time required is related to the height of the tree and the regression computation at the leaf nodes.
5. RADAR and MLE methods: We have to calculate the distance or probability of the new incoming signal to each location, which can be done in $O(A)$ time. As there are L candidate locations, the time complexity for both methods is $O(AL)$.

3. A kernel is isotropic if $k(\mathbf{x}_i, \mathbf{x}_j)$ depends only on the distance $\|\mathbf{x}_i - \mathbf{x}_j\|^2$.

4. In fact, as the α vector is typically sparse, not all training samples will be involved in the computation of the canonical vectors. By eliminating the corresponding kernel evaluations, the first step can be performed much faster.

4 Experiments

In this section, we perform a series of WLAN location estimation experiments in a realistic environment shown in Fig. 2a, the office area of the Department of Computer Science, the Hong Kong University of Science and Technology. Its area is about 64m by 40m, with three entrances and four hallways. It is equipped with an IEEE 802.11b wireless network in the 2.4GHz frequency bandwidth. All data are collected with an IBM laptop computer with an external Linksys Wireless-B USB network adapter. We divide the four hallways in Fig. 2a into a total of 99 grids, with each grid measuring 1.5m $\times$ 1.5m. There are three access points shown in the figure, though a total of eight access points (including some from other floors) are detected. Each sample is thus an 8-dimensional signal strength vector, with the measurements averaged in one second. One hundred such samples are collected at the center of each grid, with a total of 9,900 samples obtained. We randomly use 65 percent of the 9,900 samples for *training* and the rest for *testing*. All data are not normalized and no other preprocessing is performed. For comparison with LE-KCCA, we also run:

1. Support vector machine (SVM) [43],
2. Support vector regression (SVR) [43],
3. Model tree [44],
4. Maximum likelihood estimation (MLE) [8], and
5. RADAR [4] (Section 2.1).

As in LE-KCCA, both SVM and SVR use the Gaussian kernel. All implementations are in C++, and experiments are performed on a 533MHz Celeron-II machine. To reduce statistical variability, results here are based on averages over 10 repetitions.

4.1 Setting the LE-KCCA Parameters

To determine the tunable parameters (w_M, ν, and w_G) in the various methods, we further split the whole *training* data set into two parts: 75 percent for tentatively building the model, while the remaining 25 percent is used as a *validation* set for evaluating the performance. We enumerate a list of values for different parameters and empirically pick up those values with which the model performs well in the *validation* set. Once parameters are determined, we recombine the two parts of data (the whole training set) to build the model and evaluate the performance in the *testing* set. In this section, we discuss the selection of the LE-KCCA parameters in more detail. There are five parameters in LE-KCCA:

- w_G in the Gaussian kernel (11),
- w_M and ν in the Matérn kernel (12),
- the regularization parameter κ in (5), and

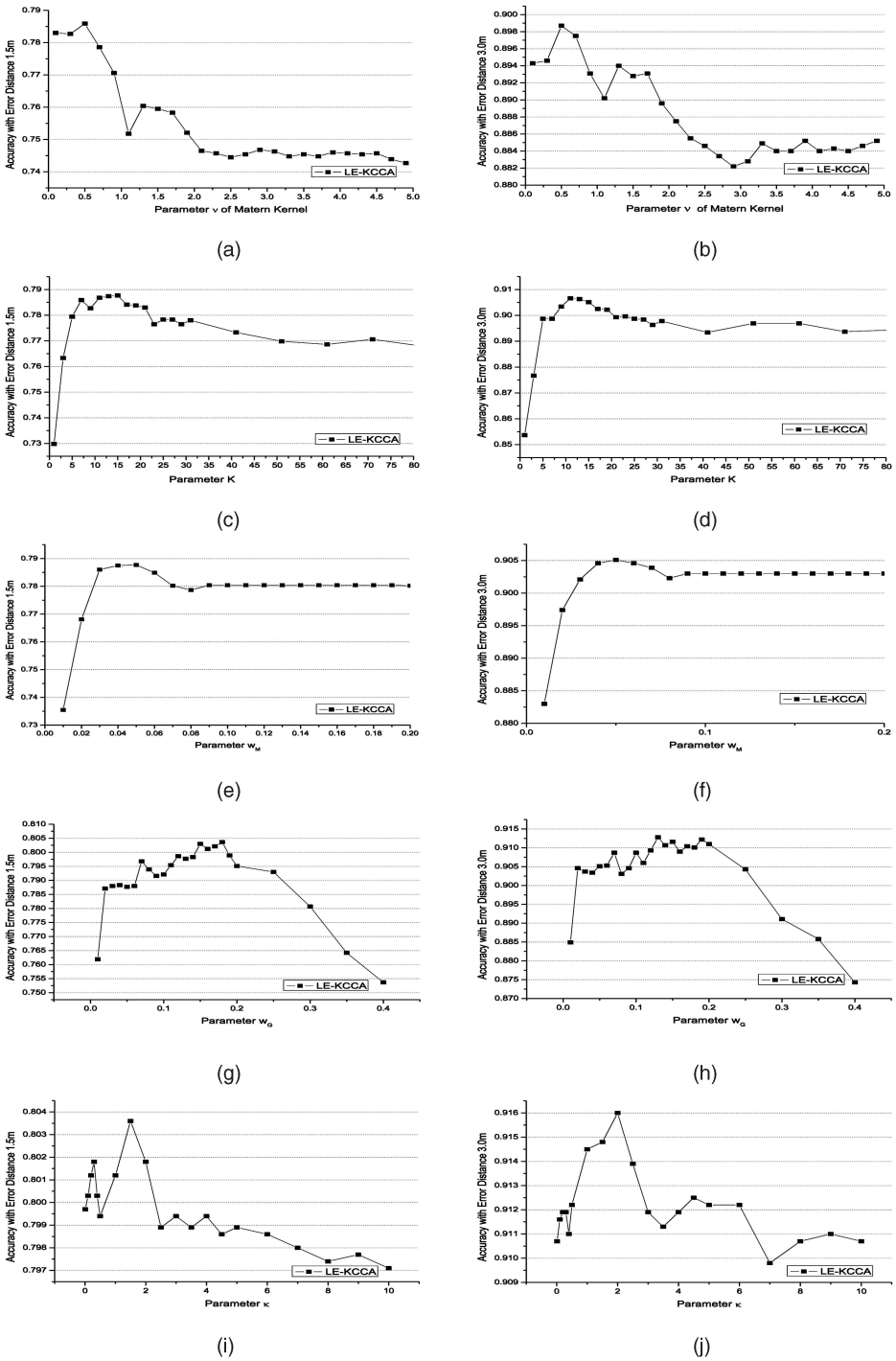


Fig. 4. Validation set accuracies at different values of ν, K, w_M, w_G and κs. (a) Different νs (error distance = 1.5m). (b) Different νs (error distance = 3.0m). (c) Different Ks (error distance = 1.5m). (d) Different Ks (error distance = 3.0m). (e) Different w_Ms (error distance = 1.5m). (f) Different w_Ms (error distance = 3.0m). (g) Different w_Gs (error distance = 1.5m). (h) Different w_Gs (error distance = 3.0m). (i) Different κs (error distance = 1.5m). (j) Different κs (error distance = 3.0m).

- the number of neighbors K in Step 3 of the online localization phase (Section 3.2.2).

Initially, we set K to 7, w_M and w_G to some random number around 0.05, and κ to some random number around 0.1. We then tune the parameters in the order of ν, K, w_M, w_G, and κ (Fig. 4).

As can be seen from Figs. 4a and 4b, the highest accuracy is obtained at $\nu = 0.5$. In this case, the Matérn kernel degenerates to the simple exponential kernel, which is more computationally efficient. Note that, in comparison to the Gaussian kernel (which corresponds to setting $\nu \rightarrow \infty$ in the Matérn kernel), the exponential kernel drops off more rapidly than the Gaussian kernel at small values of $\|\mathbf{x}_1 - \mathbf{x}_2\|$ (Fig. 3). Hence, as the physical location measurements are relatively clean, the exponential kernel is more sensitive

than the Gaussian kernel to small changes in the physical locations.

In the online localization phase, the most important parameter is K. Figs. 4c and 4d show that, when K increases from 1 to 7, the accuracy improves quickly; when K is from 7 to about 21, the accuracy stays about the same; then, the accuracy gradually decreases when K is further increased. In the following, we set $K = 15$ (an odd number, so that the median in Step 3 of the online localization phase is always well-defined). It is interesting to compare this with RADAR [4], which also uses nearest-neighbor heuristics. Unlike LE-KCCA, RADAR benefits little in varying the value of K. As discussed in Section 4.1.2 of [4], a small value of K shows minor improvement in the accuracy, while a large value leads to rapid performance degradation. This is because the RADAR neighbors in the signal space may *not* be neighbors in the physical location space. On the other hand, LE-KCCA uses location information as feedback to guide the feature extraction process so that projections of the signal and physical location spaces are maximally correlated. Consequently, neighbors in the signal projected space are usually neighbors in the physical location projected space. More discussions will be presented in Section 4.3 and Fig. 7.

The w_M parameter reflects sensitivity of the Matérn kernel to changes in the physical location $\|\mathbf{x}_i - \mathbf{x}_j\|$. When fixing ν, the Matérn kernel M (12) is a function of the term $w_M\|x_i - \mathbf{x}_j\|$ with domain $[0, +\infty)$ and range $(0, 1]$. Intuitively, the kernel M here could be viewed as a distance similarity measurement (and yet a valid kernel, positive-definite) between two locations x_i and x_j where $M = 1$ means that they are in the same location (most similar) and $M \to 0$ indicates that they are far away from each other (most dissimilar). From Fig. 3, we could also see that M drops faster with a smaller $\|x_i - x_j\|$ than with a larger value. This implies that M is likely to be sensitive to "small distance change." Here, what is "small distance change" is controlled by the scaling parameter w_M. For example, $w_M = 0$ is an extreme that $M \equiv 1$ becomes a constant so that any two points are equally similar. $w_M \to +\infty$ is another extreme that M drops sharply so that neighbor points would be dissimilar. Both of the two extremes could not well capture the distance information in the physical location space. Instead, we should choose a balanced w_M value. By observing Figs. 4e and 4f, we empirically set $w_M = 0.05$. The role of w_G in the signal space is similar. By observing Figs. 4g and 4h, we set $w_G = 0.15$.

The regularization parameter κ is used to control the flexibility of the canonical vectors. As discussed in Section 2.3.2, one can always obtain perfect correlation and, thus, uninteresting results when $\kappa = 0$. On the other hand, if κ is too large, KCCA will be overpenalized and cannot capture the correlation between signals and physical locations. By observing Figs. 4i and 4j, we choose $\kappa = 1.5$.

Note that w_G and w_M are essentially scaling factors that apply to data in signal and location spaces, respectively. Thus, when the basic unit is changed in either space, the corresponding parameter may be changed as well. For example, if we use *foot* rather than *meter* as the basic unit in location space, w_M should be rescaled by 0.30 since $1ft \approx 0.30m$.

Furthermore, w_G may be affected by the hardware difference of network adapters from different manufacturers. Although this paper does not conduct research on the effect of hardware devices, we note that [45] shows that there is approximately a linear relationship between adaptors from different hardware vendors. Thus, an alternative is to use a linear transformation as a preprocessing step, in which parameters are obtained by a small number of calibration data, for adapting to new hardware. In such a case, we don't need to change w_G. Once w_G or the linear transformation is properly set, the difference caused by new hardware would be reduced. Consequently, new data would tend to have the same characteristics as the old so that we don't need to change ν, K, and κ.

4.2 Basis Vectors Selected by PGSO

As discussed in Section 2.3.2, partial Gram-Schmidt orthogonalization (PGSO) can be used in place of the complete Cholesky decomposition to reduce the computational requirement of KCCA on large data sets. In our experiments, PGSO is applied to both the signal space and physical location space. A total of 200-300 vectors and 20-30 vectors are picked in the signal space and physical location space, respectively. As vectors are sequentially added by greedily minimizing the approximation error, they tend to spread out in the kernel-induced feature space. Fig. 5a shows the positions corresponding to the first eight vectors[5] selected in the signal space, while Fig. 5b shows those corresponding to the first eight vectors selected in the physical location space. As can be seen, they correspond to the access points in the signal space, while, in the physical location space, they correspond to the ends, corners, and centers of the hallways.

4.3 Accuracy and Speed

Fig. 6 plots the average testing accuracies and the corresponding standard deviations at different acceptable error distances. In particular, at an error distance of 3.0m, the accuracy of LE-KCCA is 91.6 percent while those of SVM, model tree, SVR, MLE, and RADAR are 87.8 percent, 88.3 percent, 89.1 percent, 86.1 percent, and 78.8 percent, respectively. When the acceptable error distance is 1.5m, the accuracy of LE-KCCA, SVM, model tree, SVR, MLE, and RADAR are 81.7 percent, 77.7 percent, 73.7 percent, 76.7 percent, 75.8 percent, and 47.3 percent. Thus, by utilizing the pairwise distance similarities in physical locations, LE-KCCA can perform better than the other methods. The SVM, by treating each (x, y) location as a class, also considers x and y together. However, it cannot utilize the information that neighboring locations should have similar signal strengths, and, thus, it is not as accurate as LE-KCCA. Note that SVR and the model tree, which treat the physical location dimensions x and y separately, can also perform relatively well at an error distance of 3.0m. However, at an error distance of 1.5m, the model tree has much degraded performance because its locally linear heuristics are not capable of capturing the nonlinearity of the signal-location mapping. Finally, RADAR does not perform well over the whole range. It is a deterministic method and cannot well adapt to the noisy characteristics of the signal.

Table 2 compares the average CPU time for predicting a new position using the various methods. As can be seen,

5. As all the candidate basis vectors are normalized to have unit norm, the first vector (in both the signal space and physical location space) has to be manually selected by the user.

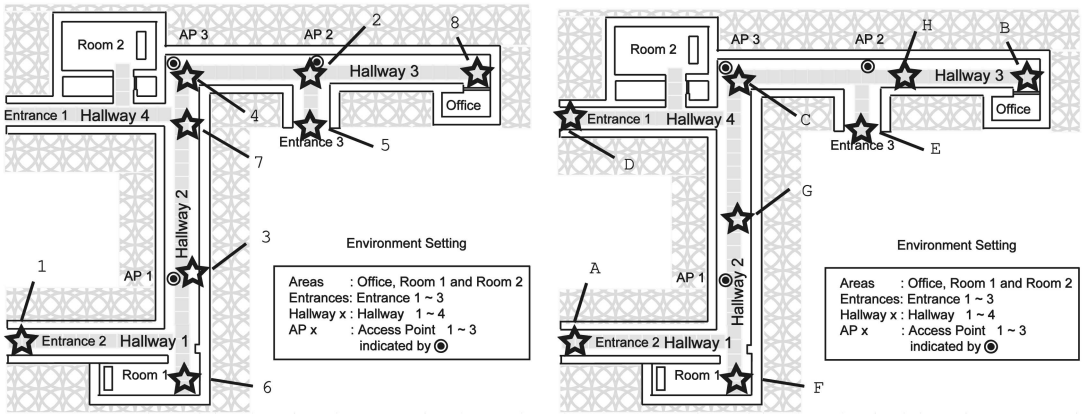


Fig. 5. Corresponding locations of the first eight vectors selected by PGSO in the signal space and the physical location space, respectively. (a) Vectors selected in the signal space (in the order $1, 2, \ldots, 8$). (b) Vectors selected in the physical location space (in the order $\mathrm{A}, \mathrm{B}, \ldots, \mathrm{H}$).

LE-KCCA is the slowest. Nevertheless, as this only takes 0.025 seconds, the online localization phase can still be performed in real-time.

With the physical location similarity as feedback, samples from the same locations move closer together, while those from different locations are pushed away under the feature-space mapping built by LE-KCCA. This is further demonstrated in Fig. 7. When we walk from hallway 1 through hallway 2 to hallway 3 (Fig. 7a), we first come closer to AP1 and then leave AP1. We then come closer to AP2 and then leave AP2. Thus, the signal strengths of AP1 and AP2 first reach their maximum and then weaken one after another (Fig. 7b). We can also see that the signal is very noisy and unstable so that neighbors in the signal space may not be neighbors in the physical location space [4]. However, after the feature mapping with KCCA, the projections in both feature spaces are maximally correlated and the trajectories become very similar to each other (Figs. 7c and 7d). Consequently, nearby physical locations have similar values in the projected signal space. In this way, even though there are not enough samples at each individual physical location, we can "borrow strength" from the nearby locations. When a new signal arrives, its nearest neighbors will be closer to the true location after the KCCA mapping than those in the original space [4], and, consequently, we can have better location estimation.

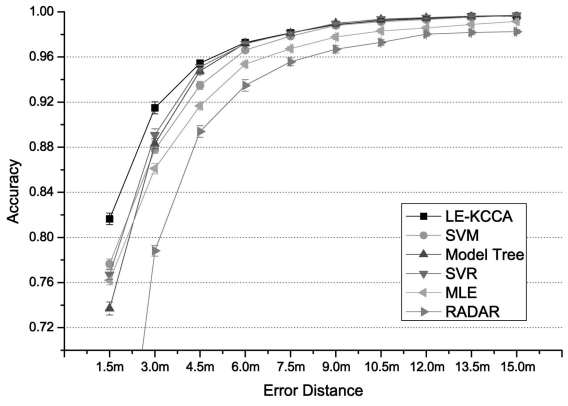


Fig. 6. Testing accuracies at different error distances.

4.4 Reduction in Calibration Effort

In the first experiment, we use all the 99 grid locations but only a random subset of the signal samples available at each location for training. Fig. 8 shows the testing accuracies at error distances of 1.5m and 3.0m. As can be seen, by using only 10-15 random training samples at each location, LE-KCCA can already outperform the other methods that use a full training set. Among the other methods, RADAR only needs to compute the average signal vector at each location and a small set of samples will suffice. Thus, its accuracy hits the (low) ceiling very quickly. In comparison, MLE can obtain a higher accuracy by learning a radio map at each location, though at the disadvantage of requiring more samples for the estimation. SVR and model tree have comparable performance with MLE. Although they utilize the location information (x, y), each dimension is treated separately and so the dependency between x and y cannot be exploited. Finally, SVM treats each physical location as an independent class label and cannot utilize the information that neighboring locations should have similar signal strengths.

In the second experiment, we select a subset of the available grid locations, and then use all the training signal samples at those selected locations for training. Evaluation is performed on both the unseen signal samples at the selected grid locations and also at the unseen locations. Again, Fig. 9 shows that LE-KCCA yields better performance.

4.5 Different Variants of LE-KCCA

As mentioned earlier, we believe that the success of LE-KCCA stems from considering information from both physical dimensions x and y together. To validate this claim, we consider the following three variants of LE-KCCA:

TABLE 2
Average CPU Time (in Seconds) for Online Localization of One New Position

method	LE-KCCA	SVM	SVR	model tree	MLE	RADAR
time	0.025	0.012	0.0084	0.00027	0.00030	0.00031

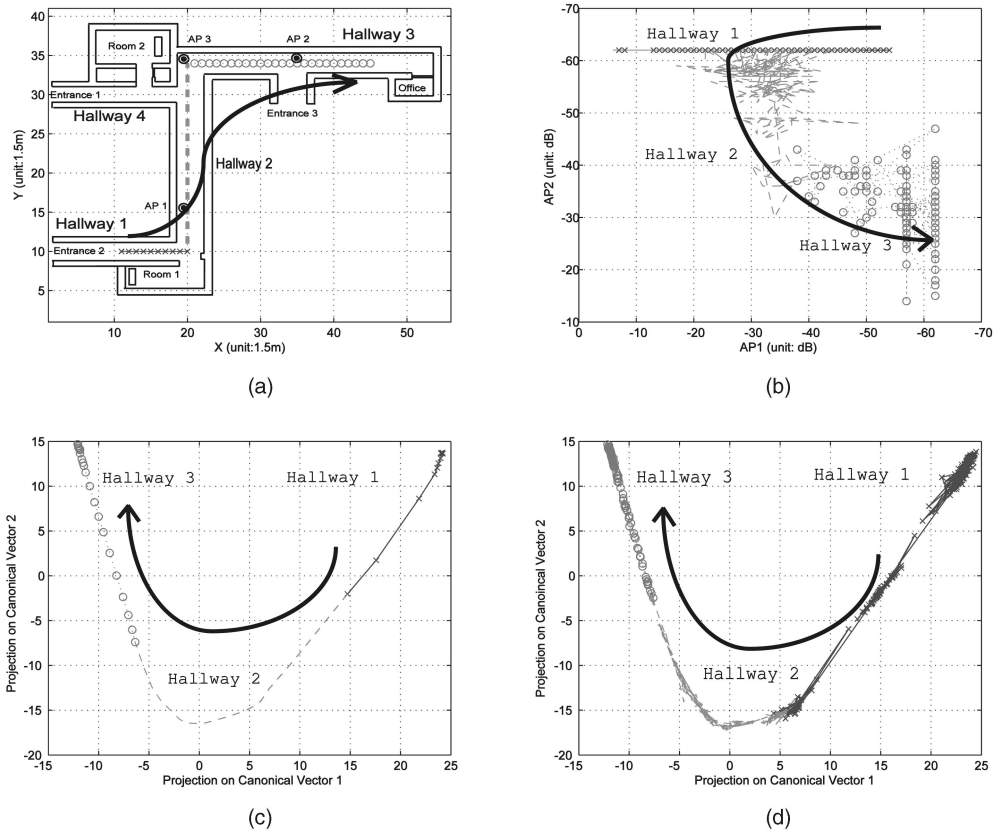


Fig. 7. Trajectories in the different spaces as one walks from hallway 1 through hallway 2 to hallway 3. Note that different sections of the trajectories are color-coded. (a) Physical location space. (b) Signal space. (c) Canonical variates in the physical location space. (d) Canonical variates in the signal space.

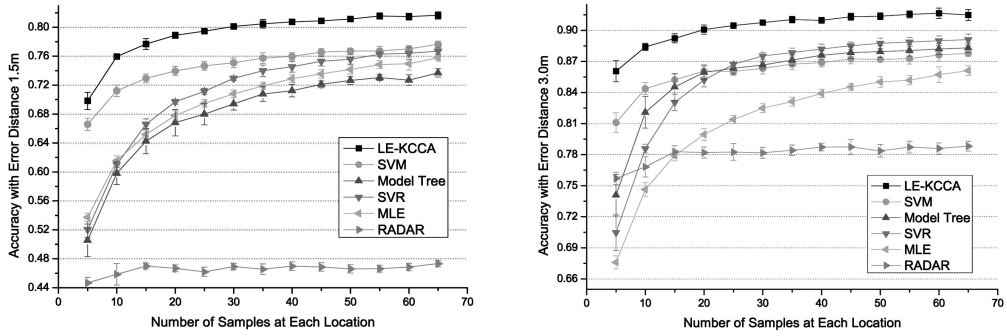


Fig. 8. Effect on varying the number of training samples at each grid location.

1. Discrete-KCCA: It treats all the physical locations independently, which is achieved by using a large value[6] for w_M in the Matérn kernel.
2. 1D-KCCA: Here, the LE-KCCA algorithm is modified so that the signal space is correlated with x and y *separately*.
3. Linear CCA: Here, instead of using the Gaussian and Matérn kernels, we use the linear kernel in both the signal and physical location spaces.

Experiments are performed in the same setting as that in Fig. 8, and results are shown in Fig. 10. As can be seen, all three variants lead to degraded performance as expected. In particular, linear CCA has an accuracy that is lower than 60 percent on every test set, and, so, its results are not shown in the figure.

6. In the experiment, we set $w_M = 50$.

5 Conclusions and Future Work

This paper focuses on using KCCA as a multidimensional vector regression method to improve the accuracy of location estimation in wireless LANs whose signal is highly uncertain, nonlinear, and correlated. Experiments show a better performance than SVR and model tree that treat each output dimension separately. We found that kernel transformation and CCA allow the construction of an accurate mapping between the physical location space and the signal space. One advantage is the higher accuracy obtained in localization with much less calibration effort. We use a

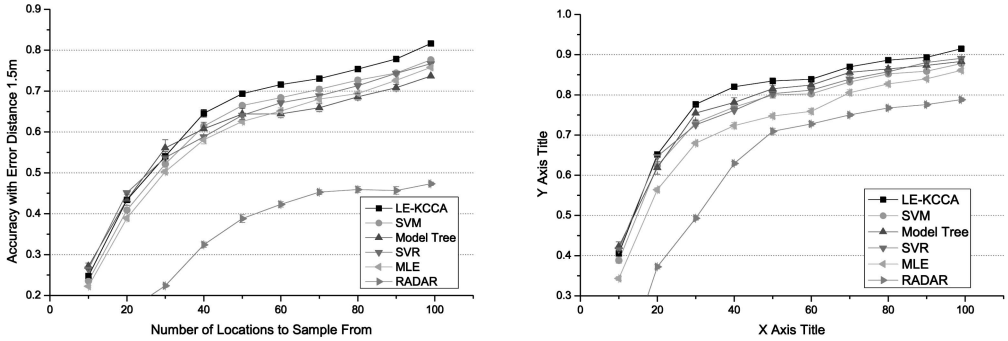


Fig. 9. Effect on varying the number of grid locations in training.

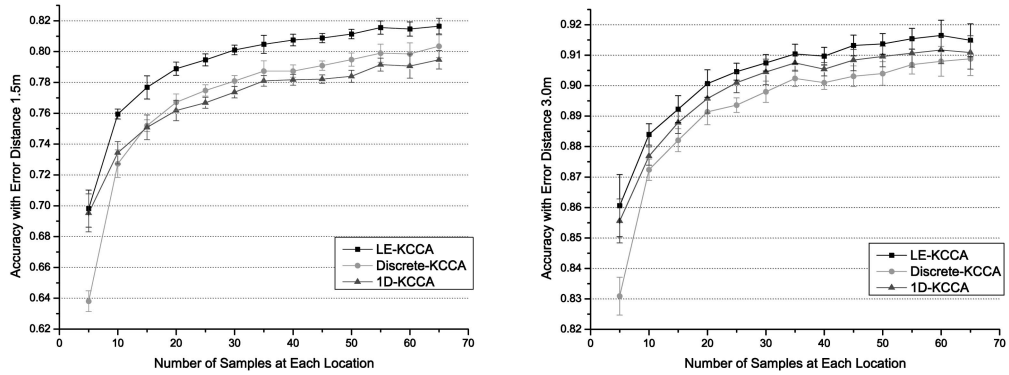


Fig. 10. The effect when the information in both physical dimensions (x and y) are *not* considered together. Discrete-KCCA considers each grid location separately, while 1D-KCCA considers x and y independently. Linear CCA has even lower accuracy and the curve is below the bottom of the figure.

Gaussian kernel for the signal space to adapt to the noisy characteristics of radio-propagation channels and a Matérn kernel for the physical location space. The advantage of KCCA is shown through extensive experimental tests in a real-world environment. The disadvantage is a slower speed when compared to the other methods, though it is still fast enough for real-time applications.

For the localization problem in this paper, we constructed the mapping without considering the additional information of user motion profiles. In the future, we plan to take the user-motion profiles into account, which may lead to higher accuracy with even less calibration effort. Similarly, we will experiment on other environmental and contextual factors in order to further boost the performance.

Note that the proposed method can be applied to a wider range of problems that have many classes but few examples. In this paper, we showed how to leverage the distance-based similarity relationship between classes to enhance classification accuracy. In the future, we will extend this for other types of interclass relationships and develop a general framework to address this new type of learning problems.

ACKNOWLEDGMENTS

The authors would like to thank Hong Kong RGC for supporting this work under grant HKUST6187/04E. They would also like to thank Jie Yin and Xiaoyong Chai for their effort in collecting the data and discussing the research problems.

REFERENCES

[1] A. Civilis, C. Jensen, and S. Pakalnis, "Techniques for Efficient Road-Network-Based Tracking of Moving Objects," *IEEE Trans. Knowledge and Data Eng.*, vol. 17, pp. 698-712, 2005.

[2] L. Liao, D. Fox, and H. Kautz, "Learning and Inferring Transportation Routines," *Proc. 19th Nat'l Conf. Artificial Intelligence*, pp. 348-353, July 2004.

[3] J. Yin, X. Chai, and Q. Yang, "High-Level Goal Recognition in a Wireless LAN," *Proc. 19th Nat'l Conf. Artificial Intelligence*, pp. 578-584, July 2004.

[4] P. Bahl and V. Padmanabhan, "RADAR: An In-Building RF-Based User Location and Tracking System," *Proc. Conf. Computer Comm.*, vol. 2, pp. 775-784, 2000, citeseer.ist.psu.edu/bahl00radar.html.

[5] C. Gentile and L. Berndt, "Robust Location Using System Dynamics and Motion Constraints," *Proc. IEEE Int'l Conf. Comm.*, pp. 1360-1364, June 2004.

[6] A. Ladd, K. Bekris, G. Marceau, A. Rudys, L. Kavraki, and D. Wallach, "Robotics-Based Location Sensing Using Wireless Ethernet," *Proc. Eighth ACM Int'l Conf. Mobile Computing and Networking*, pp. 227-238, Sept. 2002.

[7] L. Ni, Y. Liu, Y. Lau, and A. Patil, "LANDMARC: Indoor Location Sensing Using Active RFID, " *Proc. First IEEE Int'l Conf. Pervasive Computing and Comm.*, pp. 407-416, Mar. 2003.

[8] M. Youssef, A. Agrawala, and U. Shankar, "WLAN Location Determination via Clustering and Probability Distributions," *Proc. First IEEE Int'l Conf. Pervasive Computing and Comm.*, pp. 143-150, Mar. 2003.

[9] Y. Chen, Q. Yang, J. Yin, and X. Chai, "Power-Efficient Access-Point Selection for Indoor Location Estimation," *IEEE Trans. Knowledge and Data Eng.*, to appear.

[10] L. Hu and D. Evans, "Localization for Mobile Sensor Networks," *Proc. 10th Ann. Int'l Conf. Mobile Computing and Networking*, pp. 45-57, 2004.

[11] A. Savvides, C. Han, and M.B. Strivastava, "Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors," *Proc. Seventh Ann. Int'l Conf. Mobile Computing and Networking*, pp. 166-179, 2001.

[12] Y. Gwon, R. Jain, and T. Kawahara, "Robust Indoor Location Estimation of Stationary and Mobile Users," *Proc. Conf. Computer Comm.,* 2004.

[13] K. Kaemarungsi and P. Krishnamurthy, "Modeling of Indoor Positioning Systems Based on Location Fingerprinting," *Proc. IEEE Int'l Conf. Computer Comm.,* no. 1, pp. 1013-1023, 2004.

[14] D. Maligan, E. Elnahrawy, R. Martin, W. Ju, P. Krishnan, and A. Krishnakumar, "Bayesian Indoor Positioning Systems," *Proc. Conf. Computer Comm.,* vol. 2, pp. 1217-1227, Mar. 2005.

[15] T. Roos, P. Myllymaki, H. Tirri, P. Misikangas, and J. Sievanen, "A Probabilistic Approach to WLAN User Location Estimation," *Int'l J. Wireless Information Networks,* vol. 9, no. 3, pp. 155-164, 2002.

[16] P. Krishnan, A.S. Krishnakumar, W. Jun, C. Mallows, and S. Ganu, "A System for LEASE: Location Estimation Assisted by Stationary Emitters for Indoor RF Wireless Networks," *Proc. Conf. Computer Comm.,* 2004.

[17] B. Schölkopf and A. Smola, *Learning with Kernels.* Cambridge, Mass.: MIT Press, 2002.

[18] J.J. Pan, J.T. Kwok, Q. Yang, and Y. Chen, "Accurate and Low-Cost Location Estimation Using Kernels," *Proc. 19th Int'l Joint Conf. Artificial Intelligence,* to appear.

[19] S. Ganu, A.S. Krishnakumar, and P. Krishnan, "Infrastructure-Based Location Estimation in WLAN Networks," *IEEE Wireless Comm. and Networking Conf.,* Mar. 2004.

[20] A. Smailagic and D. Kogan, "Location Sensing and Privacy in a Context-Aware Computing Environment," *IEEE Wireless Comm.,* vol. 9, no. 5, pp. 10-17, Oct. 2002.

[21] E. Bhasker, S. Brown, and W. Griswold, "Employing User Feedback for Fast, Accurate, Low-Maintenance Geolocationing," *Proc. IEEE Int'l Conf. Pervasive Computing and Comm.,* pp. 111-120, Mar. 2004.

[22] X. Chai and Q. Yang, "Reducing Calibration Effort for Location Estimation Using Unlabeled Samples," *Proc. Third IEEE Int'l Conf. Pervasive Computing and Comm.,* pp. 95-104, 2005.

[23] T. Roos, P. Myllymaki, and H. Tirri, "A Statistical Modeling Approach to Location Estimation," *IEEE Trans. Mobile Computing,* vol. 1, no. 1, pp. 59-69, 2002.

[24] Z. Xiang, S. Song, J. Chen, H. Wang, J. Huang, and X. Gao, "A Wireless Lan-Based Indoor Positioning Technology," *IBM J. Research and Development,* vol. 48, nos. 5/6, pp. 617-626, 2004.

[25] A. Goldsmith, *Wireless Comm.* Cambridge: Cambridge Univ. Press, 2005.

[26] H. Hashemi, "The Indoor Radio Propagation Channel," *Proc. IEEE,* vol. 81, no. 7, pp. 943-968, 1993.

[27] J. Yin, Q. Yang, and L. Ni, "Adaptive Temporal Radio Maps for Indoor Location Estimation," *Proc. Third Ann. IEEE Int'l Conf. Pervasive Computing and Comm.,* pp. 85-94, Mar. 2005.

[28] P. Bahl, A. Balachandran, and V. Padmanabhan, "Enhancements to the RADAR User Location and Tracking System," Microsoft Research, technical report, Feb. 2000.

[29] D. Fox, J. Hightower, L. Liao, and D. Schulz, "Bayesian Filtering for Location Estimation," *IEEE Pervasive Computing,* vol. 2, no. 3, pp. 24-33, 2003.

[30] M. Youssef and A. Agrawala, "Handling Samples Correlation in the Horus Sstem," *Proc. IEEE Int'l Conf. Computer Comm.,* vol. 2, pp. 7-11, Mar. 2004.

[31] C.A. Micchelli and M. Pontil, "On Learning Vector-Valued Functions," *Neural Computation,* vol. 17, pp. 177-204, 2005.

[32] R.T.T. Hastie and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer Series in Statistics, 2002.

[33] L. Breiman and J. Friedman, "Predicting Multivariate Responses in Multiple Linear Regression," *J. Royal Statistics Soc.,* vol. 59, pp. 3-37, 1997.

[34] L. D'Ambra and R. Lombardo, "Predicting Multivariate Responses in Non-Linear Regression," *Bull. Int'l Statistical Inst.,* 1999.

[35] H. Wold, "Estimation of Principal Components and Related Models by Iterative Least Squares," *Multivariate Analysis,* P.R. Krishnaiaah, ed., pp. 391-420, New York: Academic Press, 1966.

[36] R. Rosipal and L.J. Trejo, "Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Space," *J. Machine Learning Research,* vol. 2, pp. 97-123, 2001.

[37] H. Hotelling, "Relations between Two Sets of Variates," *Biometrika,* vol. 28, pp. 312-377, 1936.

[38] D. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical Correlation Analysis: An Overview with Application to Learning Methods," *Neural Computation,* vol. 16, pp. 2639-2664, 2004.

[39] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C,* second ed. New York: Cambridge Univ. Press, 1992.

[40] M.G. Genton, "Classes of Kernels for Machine Learning: A Statistics Perspective," *J. Machine Learning Research,* vol. 2, pp. 299-312, 2001.

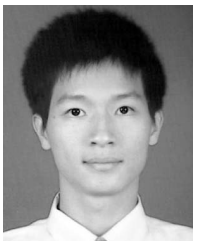
[41] M. Stein, *Interpolation of Spatial Data Some Theory for Kriging.* Springer Verlag, June 1999.

[42] A. Schwaighofer, M. Grigoras, V. Tresp, and C. Hoffmann, "GPPS: A Gaussian Process Positioning System for Cellular Networks," *Advances in Neural Information Processing Systems 16,* S. Thrun, L. Saul, and B. Scholkopf, eds. Cambridge, Mass.: MIT Press, 2004.

[43] M. Brunato and R. Battiti, "Statistical Learning Theory for Location Fingerprinting in Wireless LANs," *Computer Networks,* vol. 47, no. 6, pp. 825-845, 2005.

[44] N. Landwehr, M. Hall, and E. Frank, "Logistic Model Trees," *Proc. European Conf. Machine Learning,* pp. 241-252, 2003.

[45] A. Haeberlen, E. Flannery, A.M. Ladd, A. Rudys, D.S. Wallach, and L.E. Kavraki, "Practical Robust Localization over Large-Scale 802.11 Wireless Networks," *Proc. 10th Ann. Int'l Conf. Mobile Computing and Networking,* pp. 70-84, 2004.



Jeffrey Junfeng Pan received the bachelor's degree from the Department of Computer Science, Zhongshan (Sun Yat-sen) University in 2003, and the MPhil degree from Shanghai Jiao Tong University in 2004. He is a PhD candidate in the Department of Computer Science, Hong Kong University of Science and Technology. His research interests include data mining, machine learning, optimizationm and their applications in mobile computing and information retrieval. More information about him can be found at http://www.cs.ust.hk/~panjf.

James T. Kwok received the PhD degree in computer science from the Hong Kong University of Science and Technology in 1996. He was with the Department of Computer Science, Hong Kong Baptist University, as an assistant professor. He returned to the Hong Kong University of Science and Technology in 2000 and is now an assistant professor in the Department of Computer Science. His research interests include kernel methods, machine learning, pattern recognition, and artificial neural networks. He is serving as an associate editor for the *IEEE Transactions on Neural Networks* and the *Neurocomputing* journal. More information about him can be found at http://www.cs.ust.hk/~jamesk.

Qiang Yang received the PhD degree from the University of Maryland, College Park. He is a faculty member in the Hong Kong University of Science and Technology's Department of Computer Science. His research interests are AI planning, machine learning, case-based reasoning, and data mining. He is a senior member of the IEEE and an associate editor for the *IEEE Transactions on Knowledge and Data Engineering* and *IEEE Intelligent Systems*. More information about him can be found at http://www.cs.ust.hk/~qyang.

Yiqiang Chen received the BASc and MA degrees from the University of Xiangtan in 1996 and 1999, respectively, and the PhD degree from the Institute of Computing Technology, Chinese Academy of Sciences, in 2002. In 2004, he was a visiting scholar researcher in the Department of Computer Science at the Hong Kong University of Science and Technology (HKUST). He is now an associate professor of ICT and vice director-general of the Shanghai Branch of ICT, CAS. His research interests include data mining and mobile multimedia.