Chapter 1

Kernel Methods for RSS-based Indoor Localization

Piyush Agrawal and Neal Patwari
University of Utah, USA

Abstract: *This chapter explores the features and advantages of kernel-based localization. Kernel methods simplify received signal strength (RSS)-based localization by providing a means to learn the complicated relationship between RSS measurement vector and position. We discuss their use in self-calibrating indoor localization systems. In this chapter, we review four kernel-based localization algorithms and present a common framework for their comparison. We show results from two simulations and from an extensive measurement data set which provide a quantitative comparison and intuition into their differences. Results show that kernel methods can achieve an RMSE up to 55% lower than a maximum likelihood estimator.*

1.1 Introduction

Knowledge of user's position is becoming increasingly important in applications that include medicine and health care [1], personalized information delivery [2, 3], and security. Indoor localization algorithms have been proposed using various methods such as angle of arrival, time of flight and received signal strength (RSS), of which RSS-based algorithms are the most common.

In general, existing RSS-based indoor localization algorithms can be classified into three main categories: (1) model-based algorithms; (2) kernel-based algorithms; and (3) RSS fingerprinting algorithms. Kernel-based algorithms, the subject of this chapter, are a “middle-ground” between model-based and RSS fingerprinting algorithms.

Model-based algorithms [4, 5, 6, 7] use standard statistical channel models to provide a functional relationship between distance and RSS. Using this functional relationship, the location of a tag (unknown location device) is estimated from the RSS measured by in-range access points (APs) or anchors (known location devices) by first estimating the distances to the in-range APs using models, and then using methods of lateration to determine the coordinates. Some research [8, 9, 10] also propose using statistical models to create an entire radio map as a function of position, in which the location of the tag is estimated directly from the RSS measured by the in-range APs.

RSS-fingerprinting methods [11, 12], on the other hand, work in two phases - an *offline* training phase and an *online* estimation phase. In the offline training phase, *RF signatures* are collected at some known locations in the deployment region, which are then stored in a database. An RF signature is a vector of RSS values measured by some predetermined APs. In the online estimation phase, a location is searched from the constructed database whose RF signature matches closely with the RF signature of the tag.

Statistical channel models, in most cases, are unable to capture the complicated relationship between RSS and location in indoor environments. They also typically assume that shadow fading on links are mutually independent, even though environmental obstructions cause similar shadowing effects to many links that pass through them, an effect called correlated shadowing [13, 14].

RSS-fingerprinting methods, on the other hand, do not assume any prior relationship between RSS and position, but the training phase consumes a significant amount of time and effort [11, 15]. To some extent, the training effort can be reduced via spatial smoothing [15, 16, 17], but this is possible only to distances at which the RSS is correlated. Some research have also suggested supplementing some of the measurements using predicted RSS using channel models [11]. Changes in the environment over time reduce the accuracy of the database, requiring recalibration.

In summary, model-based algorithms require the least training effort, but they rely heavily on the prior knowledge of the relationship between RSS and position. RSS-fingerprinting algorithms, on the other hand, are not based on any prior knowledge of the relationship between RSS and position, but require considerable training effort and time.

This chapter is an exploration of kernel-based algorithms, which provide the ability to mix the features of both model-based and RSS fingerprinting algorithms. Kernel-based algorithms encapsulate the complicated relationship between RSS and position, along with correlation in the RSS at proximate locations, in a *kernel*, which can be assumed as a parametrized "black box" that takes the measured RSS as inputs and gives a *measure* of position as output. In this chapter, we describe four different kernel-based RSS localization algorithms using a common mathematical framework and compare and contrast their performance (to each other, and to a baseline model-based algorithm, the maximum likelihood estimator) using a simulation example and using an extensive experimental study. These algorithms include LANDMARC [18], Gaussian kernel localization [19], radial basis function localization [15], and linear signal-distance map localization [20].

The experimental study described in this chapter demonstrates that all four of the kernel-based localization algorithms outperform the MLE in a real-world environment. In fact, the improvement in average RMSE is as high as 55% compared to the MLE. In this chapter, we explain this improved performance of the kernel-based algorithms by using several numerical and simulation examples, in which kernel methods are shown to enable the tag's coordinate estimates to be robust to both shadowing and independent and identically distributed (i.i.d.) fading. The experimental evaluation also suggests that the complexities of the fading environment and the complicated nature of the large-scale deployment require more parameters than are available to typical model-based algorithms. In particular, in this chapter, we attempt to explain why the kernel-based algorithms perform better than model-based localization algorithms.

Standard kernel-based algorithms still require a training phase for calibration of *kernel parameters*. In this chapter, we discuss methods to minimize the calibration requirements of kernel-based algorithms by performing training simultaneously while the system is online, using pairwise measurements between APs. Specifically, several APs are deployed at some known locations throughout the building. Each AP is a transceiver and can measure the RSS of packets from other APs (although we note that we do not limit ourselves to WiFi APs; we may use any standard which allows peer-to-peer communication). These pairwise measurements constitute the training data for calibration purposes.

Outline of Chapter

Prior to discussing the four kernel methods for RSS-based localization algorithms, we present a common mathematical framework for kernel-based localization algorithms in Section 1.2.2. The remainder of Section 1.2 discusses four kernel-based algorithms. To provide more intuitive understanding of the advantages of kernel methods, we present a simple numerical example in Section 1.3. In Section 1.4, we evaluate the algorithms using kernel methods on a real-world measurement data set collected in a hospital environment. Finally we conclude this chapter in Section 1.5.

1.2 Kernel Methods

Kernel methods are a class of statistical learning algorithms in which the complicated relationship between the input (*e.g.*, signal strength) and the output (*e.g.*, physical coordinates) is encapsulated using kernel functions. A kernel function is a potentially nonlinear and parameterized function of input variables. The parameters control the functional dependencies between input and output, in our case, between signal strength and physical coordinates. A key feature of statistical learning is that it estimates the parameters based on some known input/output pairs, also called *learning* from known data. Models using kernel methods are typically *linear* with respect to the parameters, which gives them simple analytical properties, yet, are nonlinear with respect to the input variables, *e.g.*, received signal strength.

In this section, we present an overview of coordinate estimation using statistical learning with kernel methods. We begin our discussion in this section by defining our problem statement and then proceed to present a general mathematical framework for coordinate estimation using kernel methods.

1.2.1 Problem Statement

In this chapter, we consider signal strength-based tag localization. Specifically, we wish to find a two-dimensional tag coordinate, $\hat{\mathbf{x}}_t$, given the known two-dimensional reference coordinates of N APs, $\mathbf{x}_i, \forall\, i \in \{1, \ldots, N\}$, their pairwise RSS measurements, $s_{i,j}, \forall\, i \neq j,\ i, j \in \{1, \ldots, N\}$, and the RSS measured by N reference APs for a signal transmitted by a tag, $s_{i,t}, \forall\, i \in \{1, \ldots, N\}$. Also, let notation $\mathbf{s}_j$ indicate the RSS vector for AP j, where $\mathbf{s}_j = [s_{1,j}, \ldots, s_{N,j}]^T$. Similarly, let notation $\mathbf{s}_t$ indicate the RSS vector for a tag t, where $\mathbf{s}_t = [s_{1,t}, \ldots, s_{N,t}]^T$.

Note that even though we consider a two-dimensional coordinate estimation here, the same methodology can readily be extended to a three-dimensional case. Before we proceed further, we clarify our notation for the signal strength $s_{i,j}$. A measurement, $s_{i,j}$, represents the dB signal strength measured by a AP i for the signal transmitted by AP j. Similarly, subscript t indicates that the measurement is for a tag (with an *a priori* unknown location).

The measurement $s_{i,i}$, which corresponds to the RSS measured by co-located APs, is unavailable. In practice, even if two APs are located at the same position, the RSS measured between them is non-zero, *i.e.*, $s_{i,i} \neq 0$, and depends on the transmit power of the APs [20]. Some localization algorithms require the value of $s_{i,i}$ to be known; thus in this paper, we assume when necessary that $s_{i,i} = -33$ dBm [20].

We do not assume full connectivity between links. Consequently, we define set $H(j)$ to be the set of APs which are in direct communication range of AP j. Set $H(j)$ does not include the AP j and $H(j) \subset \{1, \ldots, N\}$. Similarly, $H(t)$ is the set of APs that are in direct communication range of tag t.

An AP k that is not in the set $H(j)$, is *not* in the direct communication range of AP j and would not measure any RSS from AP j. It does not necessarily mean that AP k does not receive any signal from AP j. Rather, it simply means that the signal power from AP j was so low that AP k could not demodulate its signal. This "non-measurement" of RSS by AP k is known as the "censored data" problem in statistics. We know this RSS value is low, but we do not know the value of $s_{k,j}$. How should an algorithm represent $s_{k,j}$ for $k \notin H(j)$ in its RSS vector $\mathbf{s}_j$? Most kernel-based approaches have not addressed this censored data issue, and have simply assumed full connectivity between APs. One algorithm estimates the non-measured RSS values using expectation-maximization [21]. In this article, we will provide for each kernel method a means to address non-measured RSS.

1.2.2 General Mathematical Formulation

In the framework of kernel methods, a function of the coordinate estimate of a tag, $\mathbf{f}(\hat{\mathbf{x}}_t)$, can be expressed as,

$$\mathbf{f}(\hat{\mathbf{x}}_t) = \sum_{i \in \tilde{H}(t)} \boldsymbol{\alpha}_i \phi_i(\mathbf{s}_t) + \boldsymbol{\alpha}_0 \tag{1.1}$$

where $\boldsymbol{\alpha}_i, \forall\, i \in \tilde{H}(t)$ is the *coordinate weight* of AP i, $\tilde{H}(t)$ denotes the set of APs that contribute to the kernel and $\phi_i(\cdot)$ is known as the *kernel function* corresponding to AP i. The parameter $\boldsymbol{\alpha}_0$ is known as the *bias* parameter which compensates for any fixed offset in the data [22]. In this section, we will show how the parameters, $\{\boldsymbol{\alpha}_i\}_{i \in \tilde{H}(t)}$ and $\boldsymbol{\alpha}_0$, are optimized and estimated, and how the kernel functions $\{\phi_i(\cdot)\}_{i \in \tilde{H}(t)}$ are chosen, for different algorithms and techniques in the literature.

The parameters $\{\boldsymbol{\alpha}_i\}_{i \in \tilde{H}(t)}$ are sometimes called "weights", in particular when predetermined functions are used as kernel functions $\phi_i(\cdot)$, *e.g.*, Gaussian functions. However, these parameters represent the coordinates of the APs in "location space".

Typically, these parameters are functions of the AP coordinates and are optimized to match the information given by the AP pairwise RSS measurements and their coordinates. The coordinate estimate of the tag, $\hat{\mathbf{x}}_t$, is determined by taking the inverse $\mathbf{f}^{-1}$ of $\mathbf{f}(\hat{\mathbf{x}}_t)$. Figure 1.1 shows the operation of coordinate estimation using kernel methods.

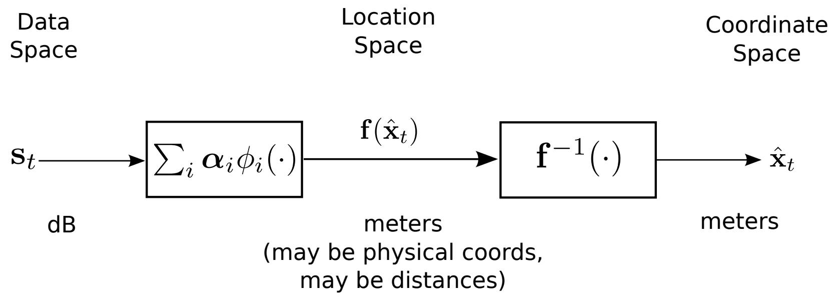

Data Space
Location Space
Coordinate Space
$\mathbf{s}_t$
$\sum_i \boldsymbol{\alpha}_i \phi_i(\cdot)$
$\mathbf{f}(\hat{\mathbf{x}}_t)$
$\mathbf{f}^{-1}(\cdot)$
$\hat{\mathbf{x}}_t$
dB
meters
(may be physical coords, may be distances)
meters

Figure 1.1: Flow chart showing the localization operation using kernel methods.

Algorithms in the class of kernel-based localization differ in the methods of optimization of $\mathbf{f}(\hat{\mathbf{x}}_t)$. Some algorithms set the kernel functions with predetermined functions and optimize the parameters $\{\boldsymbol{\alpha}_i\}_{i\in\tilde{H}(t)}$ based on pairwise RSS measurements [20, 15, 23]. In contrast, other algorithms set the parameters $\{\boldsymbol{\alpha}_i\}_{i\in\tilde{H}(t)}$ with some functions of the physical coordinates of a set of APs and optimize the kernel functions using their pairwise RSS measurements [18, 19].

Determination of kernel parameters: Typically, the kernel functions $\phi_i(\cdot)$ belong to a class of parametric nonlinear functions. Determination of kernel parameters is not a trivial task and has been extensively studied in the statistical learning literature [24]. A common technique used for their estimation is cross-validation [25]. For the purposes of cross-validation of localization algorithms, we use the data set collected between APs. In this case, the AP measurement data set is divided into two groups, one group containing $(N-1)$ APs and the other group containing one "left out" AP, where N is the total number of APs. Thus, there are N ways of dividing the data set. In cross-validation, we estimate the location of the left-out AP as if its coordinate was unknown. The location error can be determined after coordinate estimation, because every AP coordinate is, in fact, known. The average location error is computed by averaging over all left-out APs. The location error is a function of the kernel parameters. By repeating this procedure across a range of candidate values of the parameters, we can optimize the kernel parameters for the particular environment. This method is also called leave-one-out (LOO) cross-validation.

In general, existing RSS-based localization algorithms can be formulated in the framework of (1.1) by selection of:

- The function of coordinate estimates, $\mathbf{f}(\cdot)$,
- Set of APs that contribute to the kernel, $\tilde{H}(t)$,
- Coordinate weights, $\{\boldsymbol{\alpha}_i\}_{i\in\tilde{H}_t}$,
- kernel functions, $\phi_i(\cdot)$, and,
- Bias parameter, $\boldsymbol{\alpha}_0$.

In the remainder of this section, we show how the mathematical framework of (1.1) can be applied to different positioning algorithms. In particular, we select four different algorithms from the RSS-based localization literature and show how the developed framework is applied for each algorithm.

Example Framework: Consider an example wireless network with four APs deployed at known locations $\mathbf{x}_i,\, \forall\, i \in \{1,2,3,4\}$ as shown in Fig. 1.2. Also, consider a tag whose actual location (in m) is $\mathbf{x}_t = [3,2]^T$. The coordinates (in m) of the four APs are $\mathbf{x}_1 = [0.5, 0.5]$; $\mathbf{x}_2 = [0.5, 3.5]$; $\mathbf{x}_3 = [3.5, 3.5]$; $\mathbf{x}_4 = [3.5, 0.5]$. Let us assume that, all the APs are in direct communication range of the other APs and the tag, *i.e.*, $|H(t)| = |H(j)| = 4,\, \forall\, j \in \{1,2,3,4\}$. Using the known locations of the APs, their pairwise RSS measurements are generated using a log-distance path-loss model. A brief description of log-distance path-loss model is given in Section 1.3. An instance of these pairwise RSS measurements is tabulated in Table 1.1. Each row of Table 1.1 represents the RSS measured by the four APs for the signal transmitted by the corresponding device. For example, the RSS values in the first row represents the RSS measured by the deployed APs when AP-1 was transmitting. Similarly, the APs' RSS measurements for the tag are generated, an instance of which is tabulated in the last row of Table 1.1.

For the purpose of this example, the values of various parameters are tabulated in Table 1.2.

The goal in this example is to estimate the location of the tag, $\hat{\mathbf{x}}_t$, using 1.) the known location of four APs, $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$, 2.) their pairwise RSS measurements $\mathbf{s}_{i,j},\, \forall\, i \neq j,\, i,j \in \{1,2,3,4\}$, tabulated in Table 1.1, and 3.) the RSS measured by the four APs for a signal transited by the tag, $\mathbf{s}_{i,t}, \forall\, i \in \{1,2,3,4\}$, tabulated in the last row of Table 1.1.

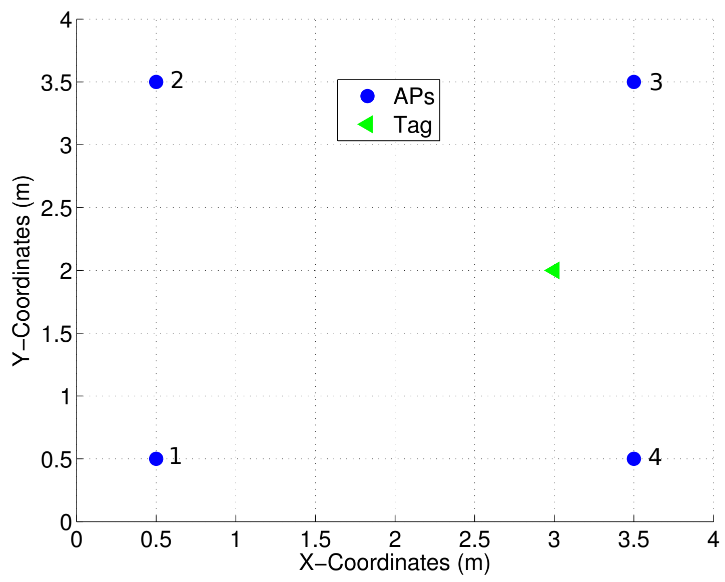


Figure 1.2: Position of APs and tags.

	Rx APs			
	AP 1	AP 2	AP 3	AP 4
Tx AP 1	n/a	-72	-62	-81
Tx AP 2	-70	n/a	-59	-85
Tx AP 3	-60	-65	n/a	-63
Tx AP 4	-59	-77	-72	n/a
Tag	-72	-69	-61	-60

Table 1.1: Table showing an example RSS values (in dBm) measured by APs.

parameter	Description	Value
n_p	Path-loss exponent	4
σ_{dB}	Fading variance	6.0 dB
Π_0	Reference Rx Power (at 1 m)	-50 dBm

Table 1.2: Log-normal path-loss parameter description and values used in the running example framework.

We will revisit this example wireless network throughout this section when we consider each localization algorithm in detail.

1.2.3 LANDMARC Algorithm

LANDMARC [18] is an RSS-based localization algorithm in which a tag's coordinate estimate is given by a weighted average of the coordinates of k closest APs that can hear the tag's transmission. In this section, we present how the LANDMARC algorithm can be expressed as a kernel method. Because LANDMARC is intuitive and can be explained with a single weighted average, it helps to demonstrate the concepts of kernel algorithms in an intuitive manner, and shows how simple a kernel-based algorithm can be.

In LANDMARC, a tag's estimated coordinate is written, using $\mathbf{f}(\hat{\mathbf{x}}_t) = \hat{\mathbf{x}}_t$, $\boldsymbol{\alpha}_i = \mathbf{x}_i$, $\boldsymbol{\alpha}_0 = \mathbf{0}$, and

$$\phi_i(\mathbf{s}_t) = \frac{1/||\mathbf{s}_t - \mathbf{s}_i||^2}{\sum_{j\in\tilde{H}(t)} 1/||\mathbf{s}_t - \mathbf{s}_j||^2} \tag{1.2}$$

Applying these relations in (1.1), we have,

$$\hat{\mathbf{x}}_t = \sum_{i\in\tilde{H}(t)} \mathbf{x}_i \frac{1/||\mathbf{s}_t - \mathbf{s}_i||^2}{\sum_{j\in\tilde{H}(t)} 1/||\mathbf{s}_t - \mathbf{s}_j||^2} \tag{1.3}$$

Here, $\tilde{H}(t)$ is the set of k APs that are closest to the tag. In LANDMARC, "closeness" is quantified by the Euclidean distance between the RSS vector of AP i, $\mathbf{s}_i$, and the RSS vector of the tag, $\mathbf{s}_t$, *i.e.*, $E_i = ||\mathbf{s}_t - \mathbf{s}_i||$. We define the vector $\mathbf{E}$ as,

$$\mathbf{E} = [E_1, \ldots, E_N]^T \tag{1.4}$$

The set of APs $\tilde{H}(t)$ in (1.2) - (1.3) is the set of k AP indices with the k smallest E_i in the vector $\mathbf{E}$. In LANDMARC, k is a variable parameter which determines the number of APs that contribute in the kernel. Any non-measured RSS in vectors $\mathbf{s}_t$ or $\mathbf{s}_i$, as discussed in Section 1.2.1, is replaced by the minimum RSS observed over the duration of the experiment minus one.

Estimation of Parameters: The only parameter that needs to be estimated in LANDMARC is the set of APs that contribute in the kernel, $\tilde{H}(t)$, which in turn depends on the parameter k. If $k = 1$, then we choose the AP which is "closest" (smallest E_i) to the tag as the coordinate estimate of the tag. Similarly, if $k = 2$, then the two 'closest' APs are considered and parameters are determined using (1.2). Unfortunately, there is no analytical solution for the optimal value of k, although it can be determined experimentally for a given environment or by using the cross-validation approach described in Section 1.2.2.

The above argument implicitly assumes that the optimal value of k is less than the number of neighbors of a tag, *i.e.*, $k < |H(t)|$, where $H(t)$ denotes the set of one-hop neighboring APs which hear the transmission from the tag. A question that arises is what would be the set $\tilde{H}(t)$ when k is greater than the number of one-hop APs, $H(t)$. A naive approach for this situation would be to place an upper threshold on the value of k. In other words, $k' = \min\{k, |H(t)|\}$, where k' is the actual number of neighbors used for a particular tag t.

Example 1.1 Revisit Example Framework
Consider the wireless network of Fig 1.2. Estimate the tag's coordinate, $\hat{\mathbf{x}}_t$, using the LANDMARC algorithm.

Solution: Let us assume $k = 3$. Using (1.4) and the definition of E_i, we can compute the Euclidean distance vector (in dBm) as,

$$\mathbf{E} = [44.36,\, 43.86,\, 30.71,\, 33.58]^T.$$

The set of APs $\tilde{H}(t)$ in (1.2) and (1.3) is the set of three AP indices with the smallest E_i in vector $\mathbf{E}$. Thus, $\tilde{H}(t) = \{2, 3, 4\}$. The values kernel function, $\phi_i(\cdot)$, corresponding to each AP $i \in \tilde{H}(t)$, is computed using (1.2),

$$\phi_2(\mathbf{s}_t) = 0.21, \quad \phi_3(\mathbf{s}_t) = 0.43, \quad \phi_4(\mathbf{s}_t) = 0.36$$

Using these kernel values and the known AP coordinates, $\{\mathbf{x}_2,\, \mathbf{x}_3,\, \mathbf{x}_4\}$, in (1.3), the LANDMARC coordinate estimate of the tag, $\hat{\mathbf{x}}_t$ (in m), is computed as,

$$\hat{\mathbf{x}}_t = [2.9,\, 2.4]^T.$$

Compared to the actual tag location, the LANDMARC coordinate estimate has an error of 0.44 m.

1.2.4 Gaussian Kernel Localization Algorithm

The Gaussian kernel localization algorithm is an RSS-based localization algorithm proposed by Kushki *et al.* [19]. Similar to LANDMARC, in the Gaussian kernel localization algorithm, the coordinate estimate of a tag is the weighted average of coordinates of the closest APs. However, the weights are determined by a Gaussian kernel, which gives a measure of distance between the RSS vector of the tag and the RSS vectors of the APs. Any stationary kernel function could be used in place of the Gaussian kernel to determine the weights. The authors chose to use the Gaussian kernel because it is widely used and studied in the literature. As presented by the authors, this algorithm uses the AP measurements over time, for time $\tau = 1, \ldots, T$, denoted by $\mathbf{s}_i^{(\tau)}$. In this section, we briefly describe the various aspects of this algorithm and how it fits into the framework developed in Section 1.2.2.

In the Gaussian kernel positioning algorithm, the coordinate estimate of the tag, $\hat{\mathbf{x}}_t$, is written as in (1.1) with $\mathbf{f}(\hat{\mathbf{x}}_t) = \hat{\mathbf{x}}_t$, $\boldsymbol{\alpha}_i = \mathbf{x}_i$, $\boldsymbol{\alpha}_0 = \mathbf{0}$, $\tilde{H}(t) = H(t)$, and,

$$\phi_i(\mathbf{s}_t) = \frac{1}{T(\sqrt{2\pi\sigma^2})^d} \sum_{\tau=1}^{T} \exp\left(\frac{-||\tilde{\mathbf{s}}_t - \tilde{\mathbf{s}}_i^{(\tau)}||^2}{2\sigma^2} \right), \tag{1.5}$$

where σ is a parameter called 'width' of the kernel. The notation $\tilde{\mathbf{s}}_t$ represents the 'reduced' $\mathbf{s}_t$, the RSS vector of the tag, and is given as, $\tilde{\mathbf{s}}_t = [s_{t,i_1}, \ldots, s_{t,i_d}]^T$, where $i_1, \ldots, i_d$ is a list of the elements in $H_d(t)$, a set of d predetermined APs. Similarly, $\tilde{\mathbf{s}}_i^{(\tau)}$ represents the reduced RSS vector for AP i at a particular time instant $\tau, \forall\, \tau \in \{1, \ldots, T\}$. The estimation of the AP set $H_d(t)$ will be discussed below in the "Estimation of Parameters" subsection.

Simplifying, (1.1) reduces to,

$$\hat{\mathbf{x}}_t = \sum_{i \in H(t)} \mathbf{x}_i \phi_i(\mathbf{s}_t). \tag{1.6}$$

The kernel functions, $\{\phi_i(\cdots)\}$, of (1.5) may be normalized [19]. Normalization avoids the situation where there are "holes" in the RSS space, regions of $\tilde{\mathbf{s}}_t$ where the sum in (1.5) has a low value. This would lead to fewer predictions at those regions of the RSS space [26]. In other words, normalization makes sure that the resulting kernel covers the whole range of RSS values measured by APs. The normalized kernel functions, $\phi_i(\mathbf{s}_t)$ can be represented as,

$$\phi_i(\mathbf{s}_t) = \frac{1}{C} \left[\frac{1}{T(\sqrt{2\pi\sigma^2})^d} \sum_{\tau=1}^{T} \exp\left(\frac{-||\tilde{\mathbf{s}}_t - \tilde{\mathbf{s}}_i^{(\tau)}||^2}{2\sigma^2} \right) \right], \tag{1.7}$$

where

$$C = \sum_{i \in H(t)} \phi_i(\mathbf{s}_t)$$

Estimation of Parameters: The Gaussian kernel localization algorithm requires estimation of the following parameters:

1. Width of the kernel, σ,

2. A predetermined set of d APs, $H_d(t)$.

Neighboring APs would report correlated RSS measurements. It is suggested that including all neighbors leads to redundancy as well as biased estimates [19]. One can minimize this effect by selecting a subset of d APs, $H_d(t)$, from the set of neighboring APs, $H(t)$, which have minimum redundancy. This set $H_d(t)$ is found to be the set which has minimum divergence. Let, $V_{i,j}$ denote a vector time series of RSS measured by AP i for a signal transmited by AP j, *i.e.*, $V_{i,j} = [s_{i,j}^{(1)}, \ldots, s_{i,j}^{(T)}]$. The set of d APs, $H_d(t)$, is selected such that,

$$H_d(t) = \underset{H_p(t)\subset H(t):|H_p(t)|=d}{\operatorname{argmin}} \sum_{h_i,h_j\in H_p(t)} |h_i - h_j|, \tag{1.8}$$

where

$$|h_i - h_j| = \min_k \; \Delta(V_{i,k} - V_{j,k}),$$

where $\Delta(V_{i,k} - V_{j,k})$ is the *divergence* between two time series $V_{i,k}$ and $V_{j,k}$ and given by [19],

$$\Delta(V_{i,k} - V_{j,k}) = \frac{1}{8}(\mu_{i,k} - \mu_{j,k})^2 \left[\frac{\sigma_{i,k} + \sigma_{j,k}}{2}\right]^{-1} + \frac{1}{2}\log\frac{0.5(\sigma_{i,k}^2 + \sigma_{j,k}^2)}{\sigma_{i,k}\sigma_{j,k}} \tag{1.9}$$

where $\mu_{i,k}$ and $\mu_{j,k}$ are the means and $\sigma_{i,k}$ and $\sigma_{j,k}$ are the standard deviations for the time series $V_{i,k}$ and $V_{j,k}$ respectively. Although other divergence measures exist in the literature, the divergence represented in (1.9) is commonly used when the distribution of the time series is Gaussian.

Another parameter to be determined is σ, the kernel width parameter. Note that the kernel width parameter, σ is a global parameter which depends on the pair-wise measurements of the APs and is independent of the RSS measurement of the tag. In our evaluation in Section 1.4, σ is determined using the cross-validation approach described in Section 1.2.2.

Example 1.2 Revisit Example Framework
Let's return to the example wireless network of Fig. 1.2. Estimate the tag's coodinate, $\hat{\mathbf{x}}_t$, using the Gaussian kernel localization algorithm.

Solution: Note that the Gaussian kernel localization algorithm uses the AP measurements over time, which is used to determine the subset of APs that have minimum redundancy in their measured RSS. Our example framework cannot provide information about this redundancy because of the pairwise i.i.d. nature of RSS simulation model used in this example. For the purposes of this example, let us assume the set $H_d(t) = \{1, 2, 4\}$.

The next parameter we need to determine is the width of the kernel, σ, which is estimated using the cross validation approach described in Section 1.2.2. Based on the experimental evaluation in Section 1.4, the value of the kernel width is found to be $\sigma = 30$ dB.

Using the RSS values from Table 1.1 along with $H_d = \{1, 2, 4\}$, $\sigma = 30$ dB, and $T = 1$ in (1.7), the values of normalized kernel functions, $\phi_i(\ldots)$, corresponding to AP $i, \forall\, i \in \{1, 2, 3, 4\}$, are determined to be:

$$\phi_1(\mathbf{s}_t) = 0.16, \quad \phi_2(\mathbf{s}_t) = 0.16, \quad \phi_3(\mathbf{s}_t) = 0.42, \quad \phi_4(\mathbf{s}_t) = 0.27.$$

Using these values of the kernel functions and their corresponding coordinates in (1.6), the estimated tag coordinate, $\hat{\mathbf{x}}_t$, (in m) is:

$$\hat{\mathbf{x}}_t = [2.6,\ 2.2]^T.$$

Compared to the actual tag coordinate, the Gaussian kernel-based coordinate estimate has an error of 0.5 m.

1.2.5 Radial Basis Function Based Localization Algorithm

In the statistical learning literature, radial basis functions have been widely used as the kernel functions $\phi_i(\cdot)$. Radial basis functions were introduced for the purpose of *exact* function interpolation. Given a set of training inputs and their corresponding outputs, the purpose of radial basis function interpolation is to create a smooth function that fits training data exactly [22].

Functions of this class have a property that the basis functions depends only on the *radial distance* (typically Euclidean) from a center $\boldsymbol{\mu}_i$, such that,

$$\phi_i(\mathbf{s}_t) = h(||\mathbf{s}_t - \boldsymbol{\mu}_i||),$$

where $h(\cdot)$ is a radial basis function. Typically, the number of basis functions and the position of their centers, $\boldsymbol{\mu}_i$, are based on the input data set $\{\mathbf{s}_i\}_{i=1,\ldots,N}$. A straightforward approach is to create a radial basis function centered at every $\{\mathbf{s}_i\}_{i=1,\ldots,N}$.

In the context of tag localization, the use of radial basis functions was proposed by Krumm *et al.* [15]. Primarily, the authors of [15] introduced interpolation using radial basis functions as a way to reduce the calibration effort of RADAR positioning [11] at the same time maintaining an acceptable location error. Specifically, the authors construct an interpolation function using radial basis functions that gives the location of a tag t, $\hat{\mathbf{x}}_t$, as a function of its RSS vector $\mathbf{s}_t$.

Using the same notation as in Section 1.2.1, the coordinate estimate of a tag, $\hat{\mathbf{x}}_t$, using radial basis functions can be written in our common framework using $\mathbf{f}(\hat{\mathbf{x}}_t) = \hat{\mathbf{x}}_t$, $\tilde{H}(t) = 1, \ldots N$, along with

$$\boldsymbol{\alpha}_0 = \frac{1}{N}\sum_{j=1}^{N} \mathbf{x}_j, \tag{1.10}$$

and,

$$\phi_i(\mathbf{s}_t) = \exp\left(\frac{-||\mathbf{s}_t - \mathbf{s}_i||^2}{2\sigma_{RBF}^2}\right), \tag{1.11}$$

where N denotes the total number of deployed APs, and σ_{RBF} is the width of the kernel. Applying these relations in (1.1), we have,

$$\hat{\mathbf{x}}_t = \sum_{i=1}^{N} \boldsymbol{\alpha}_i \phi_i(\mathbf{s}_t) + \boldsymbol{\alpha}_0, \tag{1.12}$$

The parameters $\{\boldsymbol{\alpha}_i\}_{i\in\{1,\ldots,N\}}$ are estimated. Unlike LANDMARC and the Gaussian kernel positioning algorithm, in which $\boldsymbol{\alpha}_i = \mathbf{x}_i$ is the *actual* AP location, in the radial basis function localization algorithm, the parameters $\{\boldsymbol{\alpha}_i\}_{i\in\{1,\ldots,N\}}$ are "artificial" coordinates for each AP set such that $\hat{\mathbf{x}}_t$ in (1.12) minimizes the location error for all training measurements.

Estimation of Parameters: There are two parameters that need to be estimated in the radial basis function based localization algorithm:

- Coordinate weights, $\{\boldsymbol{\alpha}_i\}_{i\in\{1,\ldots,N\}}$, and,
- Width of the radial basis function kernel, σ_{RBF}.

We begin with the estimation and optimization of coordinate weights $\{\boldsymbol{\alpha}_i\}_{i\in\{1,\ldots,N\}}$. Specifically, the coordinate weights $\{\boldsymbol{\alpha}_i\}_{i\in\{1,\ldots,N\}}$ are the coordinates in "location space". There is no need to make them equal to the coordinates of APs. In radial basis function localization, these parameters are optimized such that the information given by the AP pairwise RSS measurements best matches the known AP locations.

Substituting the AP pairwise RSS measurements and their corresponding coordinates in (1.12) and expressing them in matrix, we get,

$$Z = \mathbf{\Phi}\mathbf{A}, \tag{1.13}$$

where $\mathbf{A}$ is the coordinate weight matrix $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_N]^T$, and Z is a matrix whose i^{th} row, $\mathbf{z}_i$, is given as,

$$\mathbf{z}_i = [\mathbf{x}_i - \boldsymbol{\alpha}_0]^T,$$

and $\mathbf{\Phi}$ is the kernel design matrix whose i, j element, $\mathbf{\Phi}(i,j)$, is given as,

$$\mathbf{\Phi}(i,j) = \phi_i(\mathbf{s}_j) = \exp\left(\frac{-||\mathbf{s}_j - \mathbf{s}_i||^2}{2\sigma_{RBF}^2}\right) \tag{1.14}$$

An optimal solution can be found by using the method of *least-squares* [27]. Within the framework of least squares, the coordinate weight matrix, $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_N]^T$, can be estimated as,

$$\mathbf{A} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T Z, \tag{1.15}$$

The term

$$\mathbf{\Phi}^\dagger = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T$$

in (1.15) is known as the *pseudo-inverse* of the matrix $\mathbf{\Phi}$. The psuedo-inverse is a generalized matrix inverse for non-square matrices [28].

The other parameter that needs to be estimated is the radial basis function kernel width, σ_{RBF}. Estimation of this parameter, similar to the estimation of kernel width for Gaussian kernel position algorithm described in Section 1.2.4, is done via cross-validation.

Example 1.3 Revisit Example Framework
Consider the wireless network of Fig. 1.2. In this example, we will estimate the tag's coordinate, $\hat{\mathbf{x}}_t$, using the radial basis function based localization algorithm.

Solution: The first step is to estimate the value of parameter σ_{RBF}, which is estimated using cross-validation as explained in Section 1.2.2. Based on the experimental evaluation in Section 1.4, the value (in dB) of kernel width is found to be $\sigma_{RBF} = 30$ dB.

Using the AP pairwise RSS from Table 1.1, the kernel design matrix, $\mathbf{\Phi}$, is determined using (1.14) as,

$$\mathbf{\Phi} = \begin{bmatrix} 1 & 0.20 & 0.35 & 0.18 \\ 0.20 & 1 & 0.27 & 0.06 \\ 0.35 & 0.27 & 1 & 0.24 \\ 0.18 & 0.06 & 0.24 & 1 \end{bmatrix}$$

The next step is to determine the coordinate weight matrix, $\mathbf{A}$ using (1.15). In our example, $\boldsymbol{\alpha}_0 = [2,\, 2]^T$, using (1.10). The coordinate weight matrix $\mathbf{A}$ is determined to be

$$\mathbf{A} = \begin{bmatrix} -2.24 & -2.28 \\ -1.81 & 1.43 \\ 2.43 & 2.32 \\ 1.43 & -1.74 \end{bmatrix},$$

in which row i corresponds to the coordinates of the APs i in "location space", $\boldsymbol{\alpha}_i^T$. Using the estimated values of $\{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3, \boldsymbol{\alpha}_4\}$, the coordinate estimate of the tag according to radial basis function based localization algorithm is computed from (1.12) as,

$$\hat{\mathbf{x}}_t = [2.8,\, 2.1]^T$$

Compared to the actual tag location, the radial basis function-based coordinate estimate has an error of 0.25 m.

1.2.6 Linear Signal-Distance Map Localization Algorithm

The linear signal-distance map localization algorithm differs from the kernel-based localization algorithms discussed so far in which the physical coordinate of a device (tag/AP) is 'directly' expressed as a weighted nonlinear function of the RSS vector $\mathbf{s}_t$. In other words, the function of coordinate estimate, $\mathbf{f}(\hat{\mathbf{x}}_t)$, in (1.1) was the coordinate estimate itself, *i.e.*, $\mathbf{f}(\hat{\mathbf{x}}_t) = \hat{\mathbf{x}}_t$. In the linear signal-distance map algorithm, $\mathbf{f}(\hat{\mathbf{x}}_t)$ is the log of the distance between $\hat{\mathbf{x}}_t$ and each AP in $H(t)$, *i.e.*,

$$\mathbf{f}(\hat{\mathbf{x}}_t) = [\log ||\hat{\mathbf{x}}_t - \mathbf{x}_{i_1}||, \dots, \log ||\hat{\mathbf{x}}_t - \mathbf{x}_{i_n}||]^T$$

where $\{i_i, \dots, i_n\} = H(t)$ [20]. In this algorithm, we first estimate $\mathbf{f}(\hat{\mathbf{x}}_t)$, and then use the estimated $\mathbf{f}(\hat{\mathbf{x}}_t)$ and the known coordinates of the APs to position the tag, using methods like multilateration. Specifically, multilateration can be viewed as inverse $\mathbf{f}^{-1}$

function. In this section, we present a mathematical formulation of this algorithm in the framework of kernel methods developed in Section 1.2.2.

A key aspect of this localization algorithm is determing the relationship between the pair-wise RSS measurements between the APs and their geographical distances. Let the log-distance estimate between a tag, t, and AP, k, be represented by $\delta_{k,t}$, such that the estimated log-distance vector of the tag to the in-range APs, $H(t)$, be represented by $\boldsymbol{\delta}_t = [\delta_{1,t}, \ldots, \delta_{|H(t)|,t}]$. Following the same notation of Section 1.2.1 and using (1.1), the estimated log-distance vector of the tag, $\hat{\boldsymbol{\delta}}_t$, is given by,

$$\mathbf{f}(\hat{\mathbf{x}}_t) = \hat{\boldsymbol{\delta}}_t = \sum_{i=1}^{N} \boldsymbol{\alpha}_i \phi_i(\mathbf{s}_t), \tag{1.16}$$

where $\boldsymbol{\alpha}_i \in \mathbb{R}^{|H(t)|}$, N denotes the total number of deployed APs, $\tilde{H}(t) = \{1, \ldots, N\}$ and,

$$\begin{aligned} \boldsymbol{\alpha}_0 &= \mathbf{0}, \\ \phi_i(\mathbf{s}_t) &= \mathbf{e}_i^T \mathbf{s}_t \end{aligned} \tag{1.17}$$

where $\mathbf{e}_i$ is a column vector whose j^{th} element, $\mathbf{e}_i(j)$, is given as,

$$\mathbf{e}_i(j) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{1.18}$$

Note that $\boldsymbol{\alpha}_i$ is a column vector of the same length as $\boldsymbol{\delta}_t$. Once the log-distance to every known location AP in the set $H(t)$ is estimated, the coordinate of the tag can be determined using techniques like multilateration.

From (1.16), we can observe that the log-distance estimate of between a tag and AP is a *linear* function of the raw RSS. The basis for this linearity comes from existing radio propagation models such as the log-distance path-loss model [29]. A typical log-distance path-loss model represents the received power P_r at a distance d from the transmitter as,

$$P_r = \Pi_0 - 10n \log_{10} d.$$

So, one might represent the log-distance as

$$\log_{10} d = \frac{\Pi_0 - P_r}{10n} \tag{1.19}$$

which shows that the log-distance is linear with respect to the received signal strength

P_r. Technically, (1.19) is affine while (1.16) is linear, however, (1.19) shows some motivation for the formulation of log-distance as linear with RSS.

Estimation of Parameters: In the linear signal-distance map algorithm the only parameters that need to be estimated are the coordinate weights $\{\boldsymbol{\alpha}_i\}, \forall\, i \in \{1, \ldots, N\}$. A least-squares approach is used. As in the radial basis function localization algorithm, let the coordinate weight matrix, $\mathbf{A}$, be represented as, $\mathbf{A} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_N]^T$. The least squares solution gives the estimate of coordinate weight matrix as,

$$\mathbf{A} = (S^T S)^{-1} S^T \log(D), \tag{1.20}$$

where S denotes the signal strength matrix such that,

$$S = [\mathbf{s}_{i_1}, \ldots, \mathbf{s}_{i_n}], \quad \text{where} \quad \{i_1, \ldots, i_n\} = H(t)$$

and, D is the Euclidean AP distance matrix, whose i, j element is the Euclidean distance between AP i and AP j, $||\mathbf{x}_i - \mathbf{x}_j||$, and $\log(D)$ is element-wise logarithm on elements of the matrix D.

Example 1.4 Revisit Example Framework
Consider the wireless network of Fig. 1.2. The goal of this example is to estimate the coordinate of the tag, $\hat{\mathbf{x}}_t$ using the linear signal-distance map localization algorithm.

Solution: As in the previous examples, we start with the estimation of parameters. Using known coordinates of the APs, the distance matrix D is:

$$D = \begin{bmatrix} 0 & 3.0 & 4.2 & 3.0 \\ 3.0 & 0 & 3.0 & 4.2 \\ 4.2 & 3.0 & 0 & 3.0 \\ 3.0 & 4.2 & 3.0 & 0 \end{bmatrix}.$$

Note: Typically, before taking the logarithm of matrix D, the diagonal is replaced by a small positive value ϵ, in order to avoid taking logarithm of zero. It has been recommended in [20] to take the value of $\epsilon = d_{min}/e$, where d_{min} is the minimum value of the off-diagonal elements of the matrix D. In our example, $d_{min} = 3.0$ m and, hence, $\epsilon = 1.1$ m.

The RSS matrix S in (1.20) is constructed from Table 1.1. Using the matrices S

and D, the coordinate weight matrix $\mathbf{A}$ is determined using (1.20) as,

$$A = \begin{bmatrix} -0.032 & 0.001 & 0.015 & -0.004 \\ -0.005 & -0.025 & -0.006 & 0.002 \\ 0.014 & 0.003 & -0.031 & 0.009 \\ -0.006 & 0.006 & -0.006 & -0.021 \end{bmatrix},$$

in which row i corresponds to the optimized coordinates of AP i in the "data space", $\boldsymbol{\alpha}_i^T$. Using the estimated values of $\{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3, \boldsymbol{\alpha}_4\}$ in (1.16), the estimated log-distance of the tag to the four APs, $\hat{\boldsymbol{\delta}}_t$ is:

$$\hat{\boldsymbol{\delta}}_t = [1.48,\ 1.11,\ 0.75,\ 0.84]^T.$$

Using this estimated log-distance to the APs, the coordinate estimate of the tag is determined using methods of multilateration, which are discussed in the other chapters of this book. Instead, in this example, we use a simple grid search method in which distances are computed between each grid point and the four APs. The grid point which gives the least squared error with the estimated distances to the four APs, $\hat{\boldsymbol{\delta}}_t$, is the desired coordinate of the tag. Using this method, the coordinate estimate of the tag is computed as,

$$\hat{\mathbf{x}}_t = [3.5,\ 2.1]^T$$

Compared to the actual tag coordinate, the linear signal distance map-based coordinate estimate has an error of 0.51 m.

1.2.7 Summary

This section first presented a mathematical framework for localization using kernel methods. Next, we showed through four RSS-based localization algorithms, how the common framework, represented in (1.1), can be applied to different positioning algorithms. The various parameters and their differences and similarities are presented in Table 1.3.

1.3 Numerical Examples

For the purposes of obtaining an intuitive understanding of the advantages of kernel methods in localization algorithms, we show some numerical examples. Specifically,

	LM	GK	RBF	SDM
$\mathbf{f}(\hat{\mathbf{x}}_t)$	$\hat{\mathbf{x}}_t$	$\hat{\mathbf{x}}_t$	$\hat{\mathbf{x}}_t$	$\hat{\boldsymbol{\delta}}_t$
$\boldsymbol{\alpha}_i$	Actual AP coords, $\mathbf{x}_i$	Actual AP coords, $\mathbf{x}_i$	Optimized Coords in “Loc. space”	Optimized Coords in “Data space”
$\phi_i(\mathbf{s}_t)$	Unitless weight determined by (1.2)	Unitless weight determined by (1.5)	Unitless weight determined by (1.11)	RSS (dBm) from tag to AP i
$\boldsymbol{\alpha}_0$	$\mathbf{0}$	$\mathbf{0}$	Centroid of all deployed APs	$\mathbf{0}$
$\tilde{H}(t)$	Top k APs	All APs	All APs	All in-range APs
length of $\mathbf{s}_t$	N	d using (1.8)	length N	length N

Table 1.3: Table summarizing the similarities and dissimilarities of LANDMARC (LM), Gaussian kernel (GK), radial basis function (RBF) and linear signal-distance map (SDM) localization algorithms.

we compare the coordinate estimation of kernel-based localization algorithms, in an example setting, with that of a maximum likelihood coordinate estimation (MLE) using the log-normal shadowing model. Before we proceed with the example, we briefly describe the MLE algorithm.

1.3.1 Maximum Likelihood Coordinate Estimation

A commonly used statistical model for radio propagation is the log-distance path-loss model, in which the shadowing is modeled as log-normal (*i.e.*, Gaussian if expressed in dB). Within this model, the dB RSS between devices i and j is represented as [29],[30],

$$s_{i,j} = \Pi_0 - 10 n_p \log_{10} ||\mathbf{x}_i - \mathbf{x}_j|| + X_{i,j} \tag{1.21}$$

where Π_0 represents the RSS at a reference distance of one meter, n_p represents the path-loss exponent, and $X_{i,j}$ (in dB) is the fading error, modeled as a Gaussian random variable with a standard deviation (in dB) of σ_{dB}.

Estimating Coordinate from RSS: Given the path-loss model parameters in (1.21), the coordinate of the tag can be estimated by maximizing the likelihood of $\mathbf{s}_t$ or minimizing the negative log-likelihood. The negative log-likelihood of $\mathbf{s}_t$ is given by (1.22),

$$L(\mathbf{s}_t|\mathbf{x}_t, n_p, \Pi_0) = C + \frac{1}{2\sigma_{dB}^2} \sum_{j \in H(t)} \left[s_{t,j} - (\Pi_0 - 10 n_p \log_{10} ||\mathbf{x}_t - \mathbf{x}_j||)\right]^2 \tag{1.22}$$

where C is a constant. Note that the negative log-likelihood equation (1.22) assumes that the shadowing on links are mutually independent. This is a simplifying assumption as it has been observed that geographically proximate links exhibit correlated shadowing [13].

The maximum likelihood coordinate estimate of the tag, $\hat{\mathbf{x}}_t^{MLE}$, can be determined by minimizing the negative log-likelihood function of (1.22),

$$\hat{\mathbf{x}}_t^{MLE} = \underset{\mathbf{x}_t}{\operatorname{argmin}} \sum_{j \in H(t)} \left[s_{t,j} - [\Pi_0 - 10 n_p \log_{10} ||\mathbf{x}_t - \mathbf{x}_j||] \right]^2 \tag{1.23}$$

In the likelihood equation of (1.22), the path-loss parameters n_p and Π_0 were assumed to be known. These parameters can be estimated using the pair-wise RSS measurements between APs and their known locations. Specifically, a linear regression is performed on the pair-wise RSS measurements and log-distances, computed using the known locations, to give the path-loss parameters n_p and Π_0 [31].

Implementation Details: Due to the lack of an analytical solution to (1.23), the minimum is computed by using a *brute-force* grid search over possible coordinates of $\mathbf{x}_t$. The deployment area is divided into a grid of predetermined size, which determines the resolution of the coordinate estimate. At each grid point the value of negative log-likelihood, $L(\mathbf{s}_t|\mathbf{x}_t, n_p, \Pi_0)$, is determined by using (1.22). The grid point which has the minimum negative log-likelihood is the MLE coordinate estimate of the tag, $\hat{\mathbf{x}}_t^{MLE}$.

The maximum likelihood coordinate estimation of (1.23) suffers from computational disadvantage. Compared to the coordinate estimation of (1.1) using kernel methods, there is no closed form solution for minimizing the the negative log-likelihood of (1.23). Typically, for real-time implementation, one must use numerical optimization methods [31].

1.3.2 Description of Comparison Example

In this section, we illustrate through example the advantages of kernel-based position estimation over model-based estimation. Consider a simple network of four APs placed at the corners of a square with a wall separating them, as shown in Fig. 1.3. We consider two tag positions, one at the center of the network and the other at the edge. Using this AP placement, we generate RSS values using (1.21) where $X_{i,j}$ includes the wall loss if the line between transmitter and receiver crosses through the wall. Specifically, the RSS, $s_{i,j}$, between devices i and j is computed using (1.21) with $n_p = 2$ and $X_{i,j}$

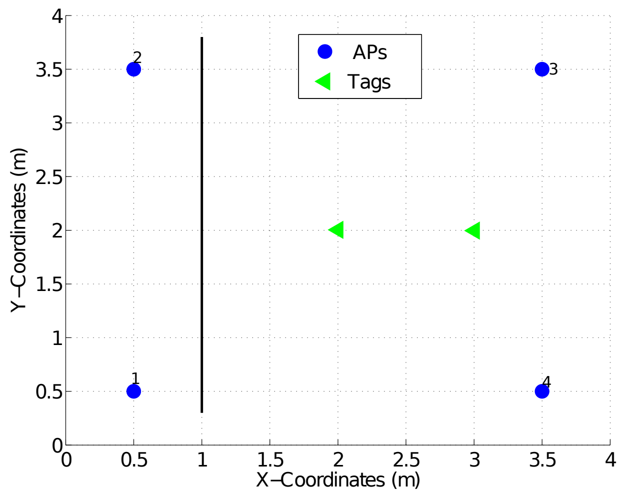


Figure 1.3: Position of APs and tags. There is a wall separating the network, represented by a black vertical line.

given as,

$$X_{i,j} = \begin{cases} L_w + Y_{i,j}, & \text{if link } (i,j) \text{ passes through the wall} \\ Y_{i,j}, & \text{otherwise} \end{cases} \tag{1.24}$$

where $Y_{i,j}$ is the shadowing loss, modeled as a zero mean i.i.d. Gaussian in dB random variable, *i.e.*,

$$Y_{i,j}[dB] \sim \mathcal{N}(0, \sigma_{dB}^2)$$

and L_w is the additional loss incurred when passing through a wall. The channel parameters used for generating the RSS vectors are tabulated in Table 1.4.

parameter	Description	Value
n_p	Path-loss exponent	2
L_w	Loss across wall	5.0 dB
σ_{dB}	Fading std. deviation	0 dB and 6.0 dB
Π_0	Reference RX Power	-40 dBm

Table 1.4: Parameter description and values used in the simulation of network. The values are assumed to be *a priori* unknown to the localization algorithm.

Note that the channel parameters given in Table 1.4 are assumed to be *a priori* unknown to the MLE algorithm. The path-loss parameters, n_p and Π_0 in (1.23), are estimated using the pairwise RSS measurements between the APs and their known locations. Specifically, a linear regression is performned on the pair-wise RSS measurements and log-distances, determined using the known locations of the APs [31, 32]. In determining the path-loss, the most recent RSS measurement between the AP pair is used.

Example 1.5
Consider the scenario when the shadowing standard deviation, $\sigma_{dB} = 0$. A noise variance of zero dB is practically not possible, but, nevertheless, it helps in understanding the effects of shadowing on coordinate estimation, with no other other fading losses. Links that pass through the wall suffer an additional loss of L_w dB due to transmission through the wall.

Solution: The coordinate estimates for different kernel algorithms discussed in the previous section are shown in Fig. 1.4. In addition to kernel-based algorithms, we plot the coordinate estimate using MLE, described in Section 1.3.1. For a fair

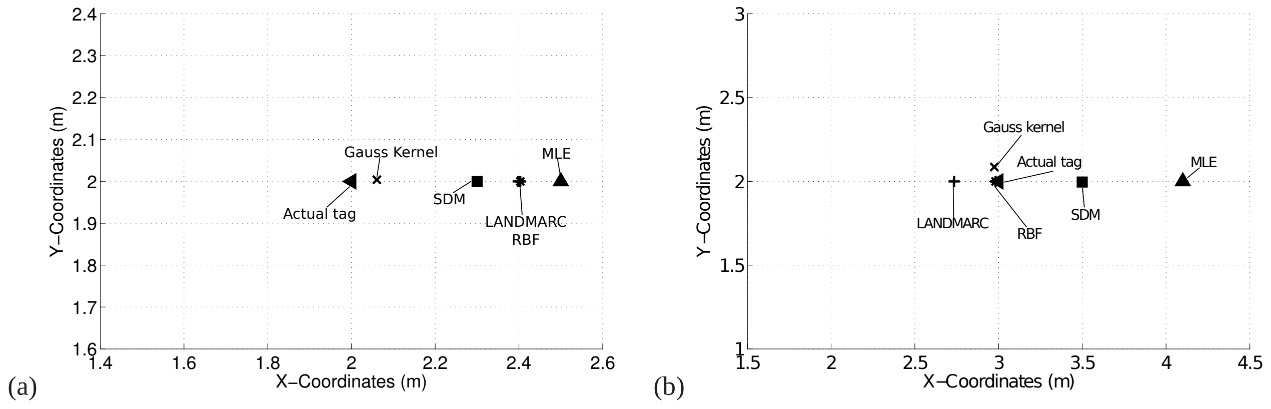


Figure 1.4: Plot showing the coordinate estimates for different localization algorithms along with the position of the tag. In (a) the tag is at the center of the network and in (b) the tag is at the edge of the network.

comparison between algorithms, we keep the set of the APs that contribute to the kernel, for different algorithms, the same and equal to four, *i.e.*, $|\tilde{H}(t)| = 4$ for all algorithms.

We also studied the performance of the coordinate estimation algorithms when different sets of APs contribute to the kernel. We observed that 1.) the Gaussian kernel and 2.) the radial basis function based algorithms have the best performance when all the in-range APs contribute to the kernel. However, for the LANDMARC localization algorithm, the best AP set $\tilde{H}(t)$ depends on the relative location of the tag in the network. For tags located at the center of the network, (*e.g.*, Fig. 1.4(a)), taking all in-range APs as the set $\tilde{H}(t)$ performs best. On the other hand, for tags located on the edge of the network, taking the top three APs as the set $\tilde{H}(t)$ performs the best.

We observe that the kernel-based algorithms for coordinate estimation perform better compared to the MLE. One can also observe that the MLE coordinate estimates have errors in the direction away from the wall. This is intuitive, because the presence of a wall between an AP and a tag, would lower the RSS of the transmitting tag. Consequently, the log-normal propagation model would predict that the tag is further away from the AP, which is behind the wall. For example, in Fig. 1.3, APs 1 and 2 are behind the wall for the two tag locations and these APs would think the tag is further away from them. Consequently, the coordinate estimate would point in the direction away from the wall. Statistically, the coordinate estimate of the tag is said to have a bias, pointing away from the wall. More bias analysis is performed in the next example.

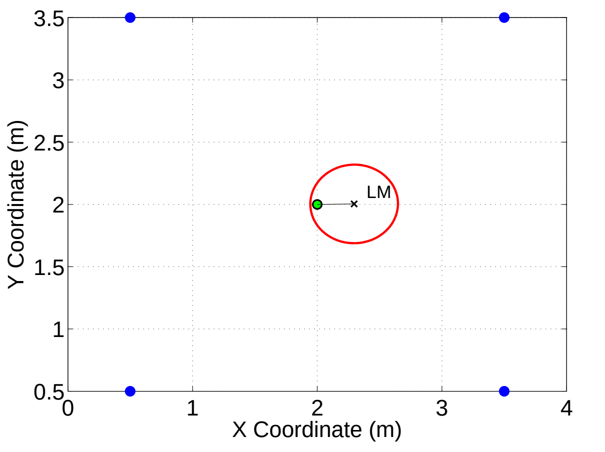
This example clearly demonstrates the effect of shadowing and how kernel-based methods can overcome these effects. Specifically, shadowing due to walls or obstacles causes a reduction in RSS from the mean RSS. General propagation models, like the one in (1.21), would account this loss to the loss suffered because of distance, in order to minimize the error $X_{i,j}$. Consequently, even in the absence of noise variance, the estimates are biased in the direction away from the source of obstruction. Kernel-based algorithms, on the other hand, "learn" to adapt to this loss because they have more freedom in the parameters than a pure model-based approach and thus, overcome the limitations of the model. Kernel methods interpolate the RSS and the physical coordinate using the AP pair-wise RSS measurements and AP known coordinates.

Example 1.6

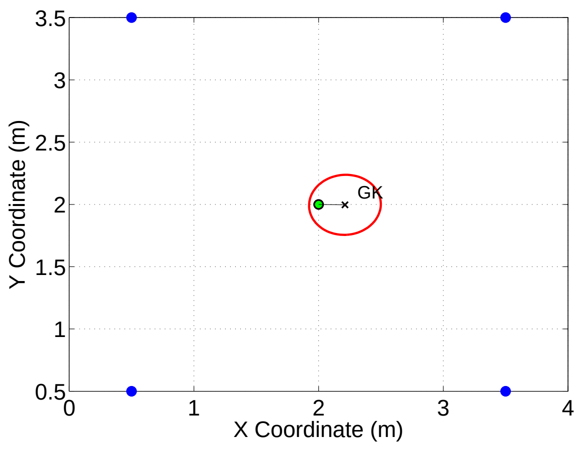
In this example, in addition to the wall loss of Example 1.5, the effect of shadowing variance is added in the path-loss equation (1.24), *i.e.*, $\sigma_{dB} > 0$. Independent Monte Carlo trials are run and the coordinate of the tag is estimated in each trial. A one standard deviation covariance ellipse and bias performance of the location estimates is then determined. The one-standard deviation covariance ellipse is a useful representation of the magnitude and variation of the coordinate estimates [33].

Solution: The covariance ellipse along with the bias performance is shown in Fig. 1.5 and Fig. 1.6. In the Fig. 1.5, the tag is located at the center of the network (at $[2, 2]^T$ m) and in the Fig. 1.6, the tag is located at the edge of the network (at $[3, 2]^T$ m).

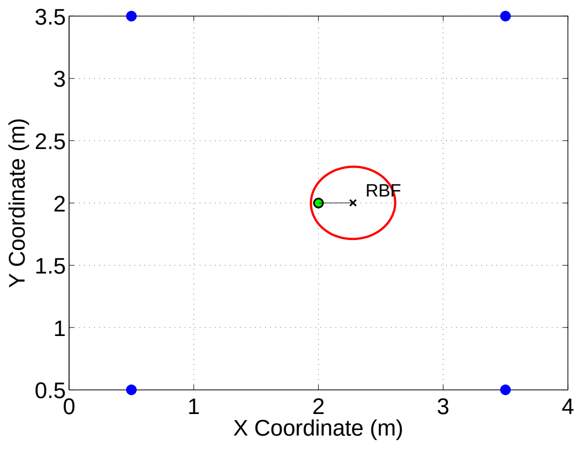
One of the most important lessons learned from this example is that the kernel-based



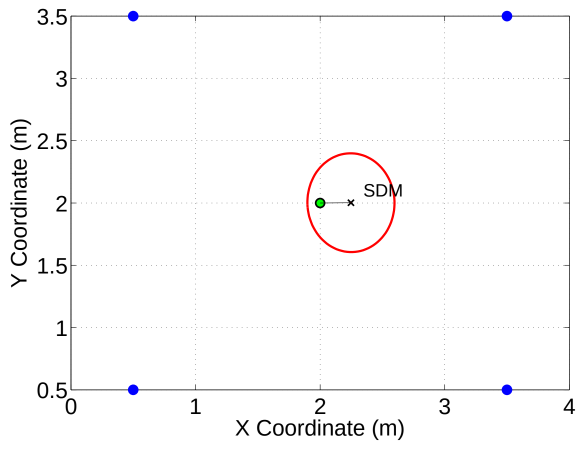
(a)



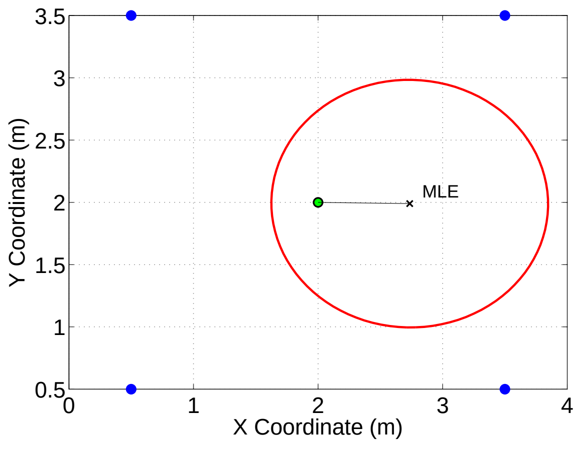
(b)



(c)



(d)



(e)

Figure 1.5: Bias plot showing the mean (×) of tag location estimates over 500 trials for LANDMARC (LM), Gaussian kernel (GK), radial basis function (RBF) and linear signal-distance map (SDM) localization algorithms. Actual tag location (●) is connected to the mean location estimate (——). Plot also shows 1-σ covariance ellipse (—) for the coordinate estimates. The APs (●) are at the corners of the grid. The tag is located at the center of the network.

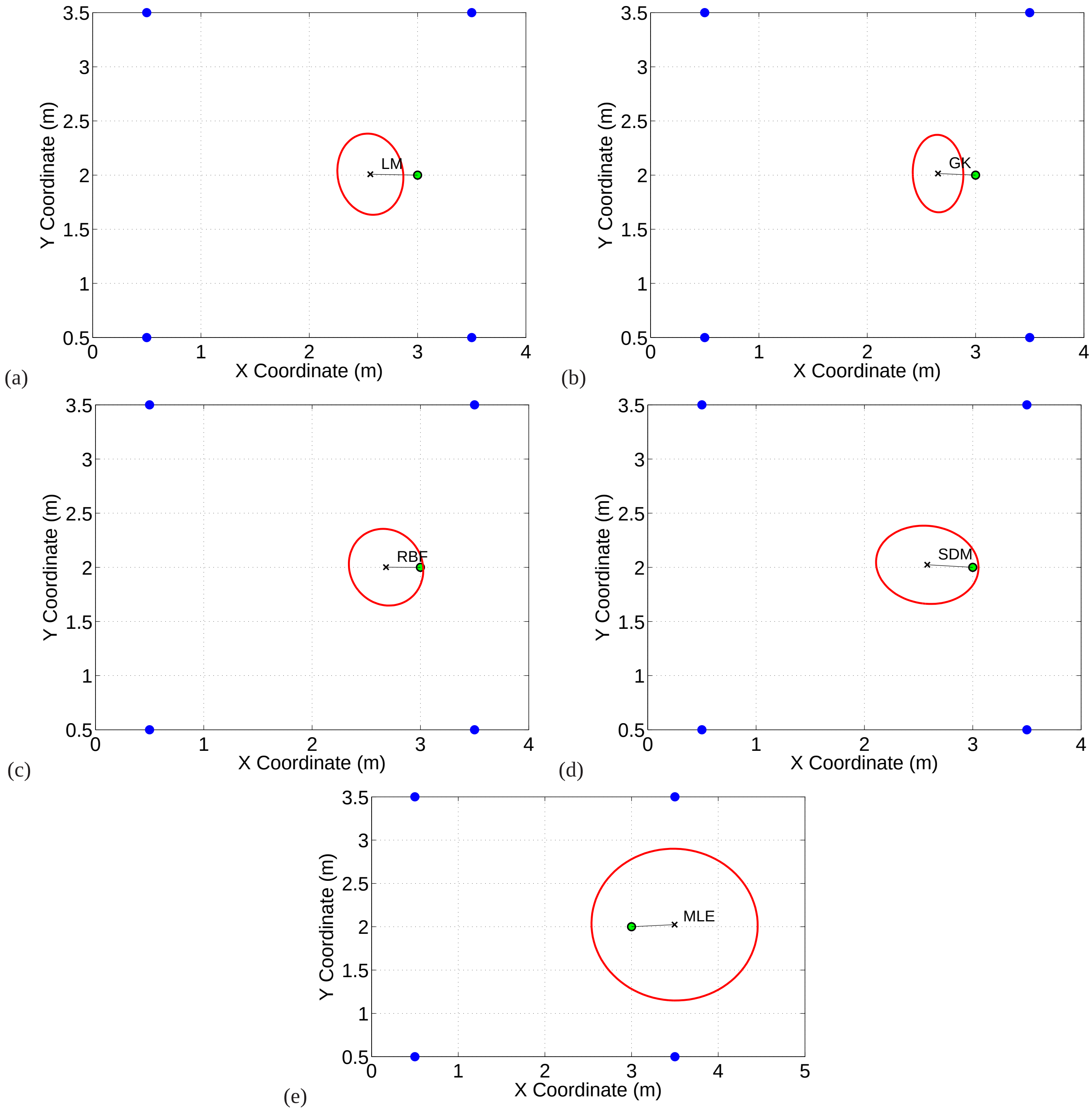


Figure 1.6: Bias plot showing the mean (×) of tag location estimates over 500 trials for LANDMARC (LM), Gaussian kernel (GK), radial basis function (RBF) and linear signal-distance map (SDM) localization algorithms. Actual tag location (●) is connected to the mean location estimate (——). Plot also shows 1-σ covariance ellipse (—) for the coordinate estimates. The APs (●) are at the corners of the grid. The tag is located at the edge of the network.

localization algorithms perform better than the MLE in terms of average RMSE. This can be easily observed in Fig. 1.5 and Fig. 1.6, where the MLE coordinate estimates, Fig. 1.5(e) and Fig. 1.6(e), suffer from both high bias and high variance. The performance improvement for kernel-based localization algorithms can be explained as follows. In the kernel-based methods, the estimated coordinate of a tag is a weighted average of some function of the coordinates of the APs that are in range of the tag. These weights are distinct for the distinct APs and each AP maintains a table of its weights for all the other APs in the network. The determination of the weights are different for different algorithms. Consequently, the APs that are on a particular side of the wall would have lower weights for the APs that are on the other side. For example, in Fig. 1.5 and Fig. 1.6, APs 1 and 2 have lower weights assigned to them by AP 3 as compared to the weight assigned to AP 4.

On the other hand, maximum likelihood coordinate estimation algorithm assumes a *common* statistical channel model for all the links in the network. Consequently, when minimizing the overall error, the path-loss exponent, which signifies the slope of the decay in RSS with respect to log-distance, is higher, similar to Example - 1. Since all the links are weighted equally, this causes a high bias in the maximum likelihood coordinate estimates, pointing away from the wall.

In summary, the advantage of the kernel-based localization algorithms over MLE is that in the kernel-based algorithms the APs which naturally have significantly different RSS values compared to the tag are weighted less compared to the other APs. On the other hand, in maximum likelihood coordinate estimation, all the in-range APs have equal weights when computing the likelihood ratio and thus, the APs which have significantly different RSS values compared to the tag dominate the coordinate estimates pushing the tag further away from its actual location.

1.4 Evaluation Using Measurement Data Set

In this section, we compare the performance of the different kernel-based localization algorithms introduced and formulated in the previous sections. Performance is quantified using two related measures:

- *Bias*: Bias is the difference between the average coordinate estimate (over many trials) and the actual coordinate. In this chapter, we show the bias using a bias plot, in which the actual coordinate and average coordinate estimate are plotted together for each tag. Bias is a consistent error in the coordinate estimate.

- *Root-mean squared error (RMSE)*: The RMSE is used to summarize both bias and variance effects. The "squared error" is the difference between the coordinate estimate and the actual location, squared, with units of m^2. The RMSE is then the square root of the average squared error (averaged over all tags in the deployment). Bias and error variance are two components of RMSE. The two together, quantified by RMSE, provide a good summary metric for quantifying localization performance.

In the rest of this section, we describe the environment along with the processing of the experimental data and the evaluation procedure for each data set.

1.4.1 Measurement Campaign Description

The measurement data consists of the pairwise RSS measured between 224 known-location wireless APs deployed on a single floor of a hospital with an area of 16,700 square meters. These APs are wireless transceivers which operate in the 2.4 - 2.48 GHz frequency band and transmit at a constant power. The APs have a limited range, and as such, the network formed by the deployed APs is not fully connected. Each AP has a limited set of neighboring APs to which it can hear and make RSS measurements. The RSS values were collected for a period of 10 minutes during which 40 RSS measurements were collected for each measurable link.

Since there are no "tags" in the measurement data, we simulate an unknown location tag using leave-one-out (LOO) procedure. In the LOO procedure, whenever we need to "create" a known-location tag, we "change" an AP into a tag for purposes of evaluation. We expect the RMSE for the leave-one-out procedure to be higher than would be seen in deployed systems with tags. APs are deployed purposefully to be spatially separated from one another, for purposes of achieving coverage with a small number of APs. So when one AP is converted to a tag, its nearest neighboring APs are relatively far from it, compared to the nearest neighbors of an actual tag that would be used in the system when no APs were "left out".

1.4.2 Evaluation Procedure

It was mentioned in Section 1.4.1 that the measurement data consists of pairwise RSS between APs only. In order to simulate a tag measurement, we employ the leave-one-out approach. As mentioned before, an AP is assumed as a tag and its position is estimated based on the remaining APs in the deployment. When we refer to a "tag" in this section, we mean the left-out AP which is used as a known-location tag.

The RMSE for each particular tag is computed based on the RSS collected over a period of 10 minutes. This procedure is repeated for all the APs in the deployment. When reporting an average RMSE, we provide two numbers. First, we average over *all* the tag (left-out AP) locations. Second, we average over just the APs in what we consider to be the "sweet spot", that is, APs in the middle of the largest section of the floor plan shown by ($*$) in Fig. 1.7. These APs should have lower bias because they are not at the edge of the building and therefore the edge of the network. The RMSE in the sweet spot provides intuition about location estimation in "good" areas, while the RMSE for all APs provides the average error result.

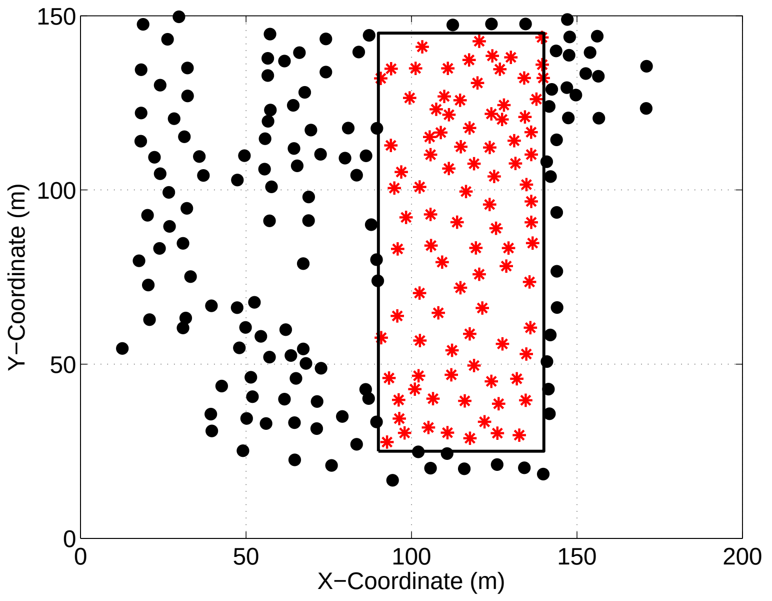


Figure 1.7: Coordinates of APs in the measurement analysis. The APs represented by ($*$) are said to be in "sweet spot" of the deployment.

1.4.3 Results

In this section, we present the results of applying the four kernel-based localization algorithms, discussed in Section 1.2, on the measurement data set. Specifically, we quantify the algorithms with the two related measures namely, 1.) bias, and 2.) root-mean squared error.

Bias Results

Figure 1.8 shows the bias plot for the four kernel-based localization algorithms and maximum-likelihood coordinate estimation. As mentioned before, bias is a consistent error in the coordinate estimates. In addition, we compute the average bias for each localization algorithm, which is shown in Table 1.5. We observe that the lowest bias is observed for the signal-distance map localization algorithm. Specifically, the average bias is 3.72 m. The coordinate estimates are more biased for the maximum-likelihood coordinate estimation algorithm, with an overall bias of 5.41 m and 5.01 m for the sweet spot region. Moreover, as one would expect, the average bias for the APs located in the sweet spot of the deployment region is lower compared to the overall average bias.

RMSE Results

In most cases, the average RMSE provides a good metric for quantifying the localization performance. The average RMSE results for different localization algorithms are tabulated in Table 1.5. From the table, we observe that all the kernel-based localization algorithms perform better than the MLE, which is a pure model based approach. In fact, the linear signal distance map localization algorithm performs the best with an overall improvement of 37% over the MLE, while the improvement is 55% for the APs in the sweet spot region.

Algorithm	Avg. bias (m)		Avg. RMSE (m)	
	O.A.	S.S.	O.A.	S.S.
LANDMARC	5.01	3.28	5.48	4.06
Gaussian Kernel	5.25	3.30	5.87	4.04
Radial basis function	4.16	2.75	4.87	3.53
Linear SDM	3.72	2.49	4.31	3.18
MLE	5.41	5.01	6.87	7.04

Table 1.5: Table showing the overall (O.A.) and sweet spot (S.S.) performance of different localization algorithms for the real-world measurement data.

1.5 Discussion and Conclusion

This chapter explores the advantages and features of a class of statistical learning algorithms, called kernel methods, as used in RSS-based localization. Kernel methods

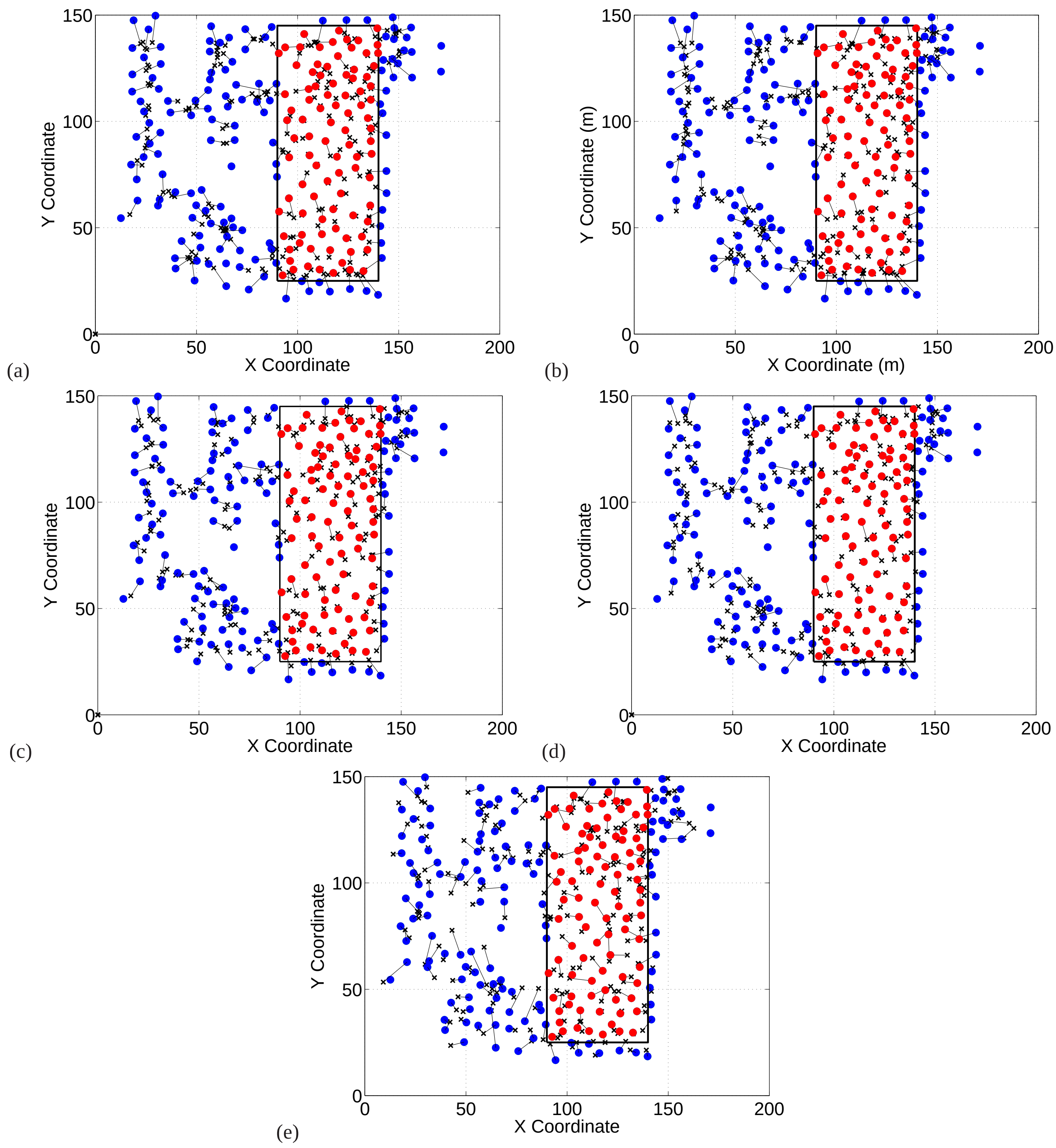


Figure 1.8: Bias plot showing the mean (×) of location estimates of the APs when it is emulated as a tag using (a) LANDMARC, (b) Gaussian kernel, (c) radial basis function, (d) linear signal-distance map, and (e) Maximum-likelihood localization algorithms. In all the plots, actual AP location (●) is connected to the mean location estimate (——). The APs in "sweet spot" are marked with (●) and are inside the box.

provide a simplified framework for localization without any *a priori* knowledge of the complicated relationship between the RSS and position. Instead, these relationships are encapsulated in parametrized nonlinear functions. Algorithms based on kernel methods inherently account for spatial correlation in the RSS, which most model-based approaches fail to capture. Kernel methods do not rely solely on a database of training measurements, like RSS fingerprinting algorithms, which must be measured very densely in space. In this chapter, a calibration scheme is presented which attempts to minimize the calibration requirements of kernel-based algorithms. Specifically, in this scheme, training is performed simultaneously while the system is online, using the AP pairwise measurements.

A simulation example of a simple four AP network is presented to provide better understanding of kernel methods. The results show that the kernel-based algorithms provide better location accuracy compared to the model-based algorithms, in terms of average RMSE. This is because kernel methods provide an adaptive weighting scheme for the APs. Within this weighting scheme, the APs that have significantly different RSS values compared to the tag are weighted less compared to the other APs.

An extensive experimental evaluation is performed for all the kernel-based algorithms and the MLE using a data set collected from a large hospital facility. These real-world experimental results indicate that all four kernel-based algorithms perform better than the MLE. In fact, the linear signal-distance map localization algorithm has the best performance in terms of average RMSE. The linear signal distance map localization algorithm has an overall RMSE reduction of 37% over the MLE, while the RMSE reduction is as high as 55% for the "sweet spot" areas of the deployment region. The complexities of the fading environment and the complicated nature of the large-scale real-world deployment require more parameters than are available to a single log-distance path-loss model. In particular, even though the linear signal-distance map localization algorithm assumes a linear with respect to log distance relationship for RSS, the parameters of the linear relationship are learned and adapted locally to the RSS measured at each AP.

Another perspective of this analysis is that spatial correlation in the RSS can be particularly useful in wireless localization. Typically, geographically proximate links would encounter similar environmental obstructions and the shadowing loss suffered on these links would be correlated. Better understanding of the area can be obtained when considering spatial correlations. Kernel methods are a strong candidate because the *kernel* in a kernel-based algorithm provides a spatial similarity measure. Additionally, kernel models are typically linear with respect to the parameters, allowing good analytical properties, yet are nonlinear with respect to the RSS measurements.

Bibliography

[1] Awarepoint, “http://www.awarepoint.com/,” 2010.

[2] M. Hazas, J. Scott, and J. Krumm, “Location-Aware Computing Comes of Age,” *Computer*, vol. 37, no. 2, pp. 95–97, 2004.

[3] G. Chen and D. Kotz, “A Survey of Context-aware Mobile Computing Research,” Tech. Rep., 2000.

[4] A. LaMarca, J. Hightower, I. Smith, and S. Consolvo, “Self-mapping in 802.11 Location Systems,” *Lecture Notes in Computer Science*, vol. 3660, p. 87, 2005.

[5] A. Savvides, C. Han, and M. Strivastava, “Dynamic Fine-grained Localization in Ad-hoc Networks of Sensors,” in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*. ACM New York, NY, USA, 2001, pp. 166–179.

[6] D. Niculescu and B. Nath, “VOR Base Stations for Indoor 802.11 Positioning,” in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*. ACM New York, NY, USA, 2004, pp. 58–69.

[7] Y. Ji, S. Biaz, S. Pandey, and P. Agrawal, “ARIADNE: A Dynamic Indoor Signal Map Construction and Localization System,” in *Proceedings of the 4th International Conference on Mobile Systems, Applications and Services*. ACM, 2006, p. 164.

[8] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievänen, “A Probabilistic Approach to WLAN User Location Estimation,” *International Journal of Wireless Information Networks*, vol. 9, no. 3, pp. 155–164, 2002.

[9] M. Youssef and A. Agrawala, “The Horus Location Determination System,” *Wireless Networks*, vol. 14, no. 3, pp. 357–374, 2008.

[10] T. King, S. Kopf, T. Haenselmann, C. Lubberger, and W. Effelsberg, "COMPASS: A Probabilistic Indoor Positioning System based on 802.11 and Digital Compasses," in *Proceedings of the 1st International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*. ACM, 2006, p. 40.

[11] P. Bahl and V. Padmanabhan, "RADAR: An In-building RF-based User Location and Tracking System," in *IEEE INFOCOM*, vol. 2, 2000, pp. 775–784.

[12] M. Brunato and C. Kalló, "Transparent Location Fingerprinting for Wireless Services," in *Proceedings of Med-Hoc-Net*, vol. 2002, 2002.

[13] P. Agrawal and N. Patwari, "Correlated Link Shadow Fading in Multi-Hop Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 8, pp. 4024–4036, 2009.

[14] N. Patwari and P. Agrawal, "Effects of Correlated Shadowing: Connectivity, Localization, and RF Tomography," in *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*. IEEE Computer Society Washington, DC, USA, 2008, pp. 82–93.

[15] J. Krumm and J. C. Platt, "Minimizing calibration effort for an indoor 802.11 device location measurement system," *Technical Report, Microsoft Coorporation*, 2003.

[16] A. Howard, S. Siddiqi, and Sukhatme, "An Experimental Study of Localization Using Wireless Ethernet," in *Field and Service Robotics*. Springer, 2006, pp. 145–153.

[17] J. Letchner, D. Fox, and A. LaMarca, "Large-scale Localization from Wireless Signal Strength," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 20, no. 1. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005, p. 15.

[18] L. Ni, Y. Liu, Y. Lau, and A. Patil, "LANDMARC: Indoor Location Sensing using Active RFID," *Wireless Networks*, vol. 10, no. 6, pp. 701–710, 2004.

[19] A. Kushki, K. Plataniotis, and A. Venetsanopoulos, "Kernel-based Positioning in Wireless Local Area Networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 689–705, 2007.

[20] H. Lim, L. Kung, J. Hou, and H. Luo, "Zero-configuration, Robust Indoor Localization: Theory and Experimentation," in *Proceedings of IEEE Infocom*, 2006, pp. 123–125.

[21] T. Roos, P. Myllymäki, and H. Tirri, "A Statistical Modeling Approach to Location Estimation," *IEEE Transactions on Mobile Computing*, pp. 59–69, 2002.

[22] C. Bishop, *Pattern Recognition and Machine Learning*. Springer New York:, 2006.

[23] Y. Gwon and R. Jain, "Error Characteristics and Calibration-free Techniques for Wireless LAN-based Location Estimation," in *Proceedings of the Second International Workshop on Mobility Management & Wireless Access Protocols*. ACM New York, NY, USA, 2004, pp. 2–9.

[24] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford Univ Pr, 2005.

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

[26] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The Elements of Statistical Learning: Data Mining, Inference and Prediction," vol. 27, no. 2, pp. 83–85, 2005.

[27] S. Kay, "Fundamentals of Statistical Signal Processing: Estimation Theory," *Prentice-Hall Signal Processing Series*, p. 595, 1993.

[28] G. Strang, "Linear Algebra and its Applications, 1988," *Hartcourt Brace Jovanovich College Publishers*.

[29] T. Rappaport, *Wireless Communication: Principles and Practice*, 2nd ed. Printice Hall, 1996.

[30] H. Hashemi, "The indoor radio propagation channel," *Proccedings of the IEEE*, vol. 81, no. 7, pp. 943–968, July 1993.

[31] N. Patwari, A. Hero, M. Perkins, N. Correal, and R. O'Dea, "Relative Location Estimation in Wireless Sensor Networks," *IEEE Transactions on Signal Processing*, vol. 51, no. 8, pp. 2137–2148, 2003.

[32] N. Patwari, Y. Wang, and R. O'Dea, "The importance of the multipoint-to-multipoint indoor radio channel in ad hoc networks," in *IEEE Wireless Communications and Networking Conference,*, vol. 2, 2002, pp. 608–612.

[33] L. Paradowski, M. Acad, and P. Warsaw, “Uncertainty ellipses and their application to interval estimation of emitter position,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 1, pp. 126–133, 1997.