5

Indoor Positioning Using Feature-Matching Methods

Previous chapters discussed range-based and geometric positioning techniques such as TOA, TDOA, and AOA methods, as well as the algorithms using estimators, filters, and optimizers. In this chapter, we focus on feature-matching methods for indoor positioning, which do not require explicit time or angle measurements. Instead, these methods leverage distinctive features or patterns observed in the environment and match them to a pre-collected database or learned model to infer location. Feature-matching approaches, sometimes categorized as "scene analysis" methods in contrast to geometric triangulation and proximity methods [1], encompass a broad category of techniques, including location fingerprinting, pattern recognition algorithms, and machine learning-based models. The common idea is to use the unique signatures of an environment, such as RF signal strengths, magnetic field anomalies, visual landmarks, or other sensor patterns as fingerprints of specific locations. During an offline phase, these fingerprints are collected and stored with their associated ground-truth locations. Then, in the online phase, a user measures the features at an unknown location and a matching algorithm attempts to find the best match in the database or model, thereby estimating the user's location.

This chapter provides a comprehensive introduction of indoor positioning using feature matching. We begin by covering the theoretical foundations and mathematical formulations of fingerprinting methods, including deterministic approaches (e.g., nearest-neighbor matching) and probabilistic approaches (e.g., Bayesian estimation). We then describe pattern recognition techniques, highlighting how classical classification and regression algorithms can be applied to indoor localization. Following that, we explore modern machine learning-based approaches, especially deep learning models that have emerged as powerful tools for building location estimators from complex and heterogeneous sensor data. We also identify key challenges unique to feature-matching methods, such as environmental dynamics, scalability of data collection, and domain adaptation and discuss how current research addresses these issues.

It is important to note that feature-matching methods are fundamentally different from the geometric approaches covered in earlier chapters. Rather than computing positions from distance or angle measurements and solving geometric equations, fingerprinting and learning-based methods treat localization as a pattern-matching or inference problem. This enables the use of readily available signals (such as Wi-Fi received signal strength) without requiring additional infrastructure, but it also introduces new challenges, which we will discuss in Section 5.4. By the end of this chapter, the reader should have a clear understanding of how indoor localization can be achieved through feature matching, the theoretical basis of these techniques, and their practical considerations.

5.1 Fundamentals of Fingerprinting for Indoor Localization

Fingerprinting is one of the most prevalent feature-matching approaches for indoor positioning. In a fingerprinting system, the environment is mapped in advance to create a fingerprint database. Each entry in this database consists of a location coordinate (often represented as a point on a floor plan) and a feature vector that captures the observable signals or patterns at that location. During the online phase, the system measures the same type of feature vector at an unknown location and compares it against the stored fingerprints to find the best match. The estimated position is then inferred to be the location associated with the most similar fingerprint in the database [2].

5.1.1 Principles of Location Fingerprinting

The basic principle of location fingerprinting is that different locations in an indoor environment often have distinguishable signatures:

- *RF signal features:* The most classic example is Wi-Fi received signal strength (RSS) fingerprinting [3]. Wireless access points (APs) are distributed around a building, and, at any given location, the set of RSS values from these APs can serve as a fingerprint. Because of attenuation by walls and distance, each location yields a different vector of RSS measurements. Similar to Wi-Fi APs, Bluetooth Low Energy (BLE) beacons broadcast signals that can be measured (for example, as RSS values or proximity estimates). A fingerprint can thus include BLE signal strengths or the set of nearby beacon identifiers, and these BLE features are often combined with Wi-Fi fingerprints to improve coverage and accuracy [4]. Channel impulse responses (CIR) or channel frequency responses (CFR) (especially from UWB or the channel state information in Wi-Fi) is an emerging area [5]. These signals could potentially improve accuracy to submeter levels by distinguishing locations based on subtle differences in channel patterns that narrowband RSS alone could not capture.
- *Magnetic field anomalies:* Indoor environments contain structural steel and electrical equipment that disturb the Earth's magnetic field. The magnetic field intensity and direction at a point can act as a fingerprint [6]. This concept has been utilized by systems that map indoor magnetic signatures and later match a user's magnetometer readings to that map.
- *Light and sound:* Characteristics of ambient light or acoustic signals (e.g., ultrasound beacons or background noise profiles) can be used in certain scenarios as location signatures.
- *Visual landmarks:* If a camera is available, visual features (such as the presence of certain markers, posters, or the scene captured by the camera) can be treated as a fingerprint of location. For example, an image taken in one hallway will have different visual features than an image taken in another hallway.
- *Heterogeneous data fusion:* Often multiple modalities are combined, such as using Wi-Fi signals together with BLE, inertial sensor patterns (such as step and turn patterns), or barometric pressure (for floor detection). These multisensor fingerprints offer higher robustness.

The fingerprinting approach generally consists of two phases, as shown in Figure 5.1. In the offline survey (also known as the training phase), a site survey is conducted where an operator (or an automated robot or crowdsourcing participants) collects data at various reference locations. At each reference

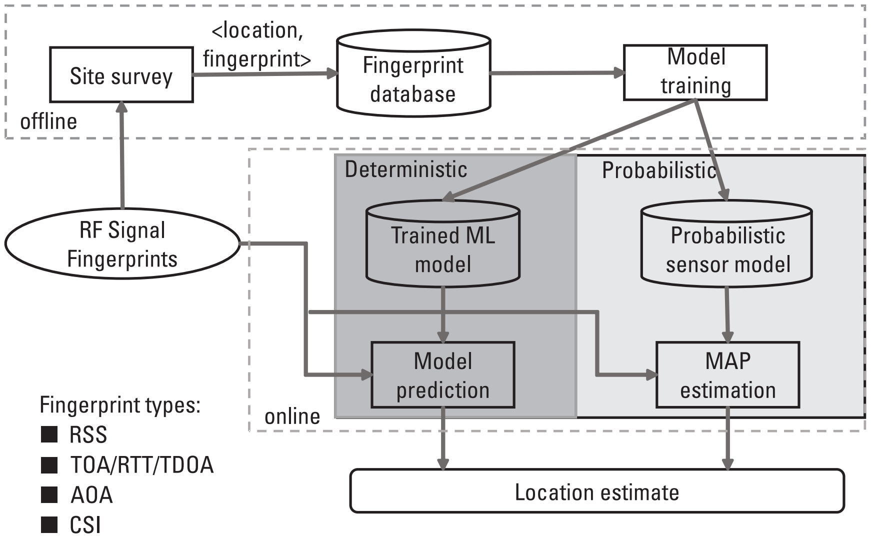


Figure 5.1 The general framework of fingerprinting-based indoor positioning with RF signals.

point (RP), the feature vector $\mathbf{f}_{\mathrm{RP}}$ is recorded along with the known coordinates $\mathbf{x}_{\mathrm{RP}}$. The set of all such pairs $\{(\boldsymbol{x}_i, \boldsymbol{f}_i)\}_{i=1}^{M}$ forms the fingerprint database, where M is the number of reference points surveyed. The offline survey also forms the correspondence between $\boldsymbol{x}_i$ and $\boldsymbol{f}_i$, which can be established either deterministically or probabilistically. Then, in the online positioning (also known as the query phase), a user at an unknown location obtains a measurement $\mathbf{z}$ (the same type of feature vector as in the fingerprints, e.g., a vector of RSS values). The system then compares $\mathbf{z}$ to all (or a subset of) stored fingerprints $\{\mathbf{f}_i\}$ and computes a similarity or distance metric. The location estimate $\hat{\mathbf{x}}$ is determined based on the best match according to this metric. This basic procedure was first demonstrated in early Wi-Fi localization systems such as RADAR by Bahl and Padmanabhan [3], which used empirical signal strength maps to achieve meter-level accuracy. A key assumption is that the environment remains relatively static between the offline and online phases so that the fingerprints remain valid over time.

5.1.2 Mathematical Formulation of the Fingerprint-Matching Problem

We can formalize the fingerprint-based positioning problem as follows. Let $\mathbf{x} \in \mathbb{R}^d$ denote the unknown true location of the user (for example, $d = 2$ for 2-D floor plan coordinates, or $d = 3$ if elevation is included). Let $\mathbf{z} \in \mathbb{R}^N$

denote the observed feature vector at that location (e.g., the vector of N signal strength readings). We assume an offline database of M reference points, with known locations $\{\mathbf{x}_i: i = 1, \ldots, M\}$ and their corresponding fingerprint feature vectors $\{\mathbf{f}_i: i = 1, \ldots, M\}$.

One straightforward approach to estimate $\mathbf{x}$ is to find the reference point whose fingerprint is closest to the observed $\mathbf{z}$ in terms of some distance metric. That is,

$$i^* = \arg\min_{1 \le i \le M} D\left(\mathbf{z}, \mathbf{f}_i\right) \tag{5.1}$$

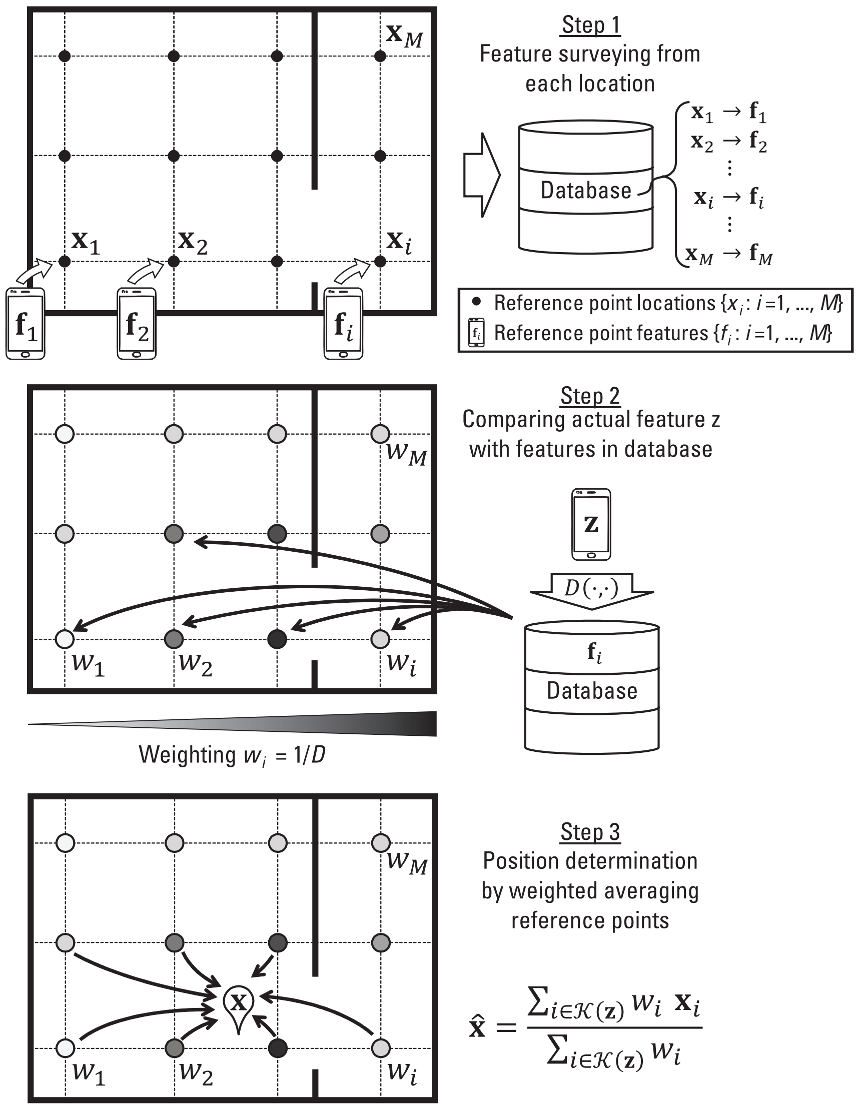
where $D(\ ,\ )$ is a distance measure such as Euclidean distance $D(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{n=1}^{N}(a_n - b_n)^2}$ or Manhattan distance $D(\mathbf{a}, \mathbf{b}) = \sum_{n=1}^{N}|a_n - b_n|$. The estimated position is then $\hat{\mathbf{x}} = \mathbf{x}_{i^*}$ (i.e., the coordinates of that nearest-neighbor reference point). This is the simplest form of deterministic fingerprint matching, often referred to as the Nearest Neighbor (NN) algorithm in pattern recognition.

In practice, using just one nearest neighbor can be susceptible to measurement noise or an outlier fingerprint. A common extension is the k-Nearest Neighbors (k-NN) approach, which finds the k closest fingerprints to $\mathbf{z}$ and then either chooses the most frequent location among them (if reference points are grouped by region) or, more commonly, computes a weighted average of their coordinates:

$$\hat{\mathbf{x}} = \frac{\sum_{i \in \mathcal{K}(\mathbf{z})} w_i \mathbf{x}_i}{\sum_{i \in \mathcal{K}(\mathbf{z})} w_i} \tag{5.2}$$

where $\mathcal{K}(\mathbf{z})$ is the index set of the k nearest reference points to $\mathbf{z}$, and w_i is a weight assigned to neighbor i. The weights w_i can be chosen, for example, as $w_i = 1/(\mathbf{z}, \mathbf{f}_i)$ (inverse distance weighting) so that closer matches have a higher influence on the final estimate. Equation (5.2) then yields an interpolated position, which can mitigate the quantization effect of returning only previously surveyed points.

Figure 5.2 illustrates the three steps to obtain a position fix using the fingerprinting method. The fingerprint-matching problem can also be cast in a probabilistic framework. Let the location $\mathbf{x}$ be modeled as a random variable and the observed fingerprint $\mathbf{z}$ as another random variable. During the offline phase, one can build statistical models or empirical histograms of $\mathbf{Z}_i$ for each reference location. For example, at reference point i, we might model the distribution of the feature vector as $P(\mathbf{z}|\mathbf{x}_i)$ (or the likelihood of {AU: Edit correct?}



Step 1
Feature surveying from each location
Database
$\mathbf{x}_1 \rightarrow \mathbf{f}_1$
$\mathbf{x}_2 \rightarrow \mathbf{f}_2$
$\mathbf{x}_i \rightarrow \mathbf{f}_i$
$\mathbf{x}_M \rightarrow \mathbf{f}_M$
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_i$ $\mathbf{x}_M$
$\mathbf{f}_1$ $\mathbf{f}_2$ $\mathbf{f}_i$
● Reference point locations $\{x_i : i = 1, ..., M\}$
Reference point features $\{f_i : i = 1, ..., M\}$
Step 2
Comparing actual feature z with features in database
$\mathbf{z}$
$D(\cdot,\cdot)$
$\mathbf{f}_i$
Database
w_1 w_2 w_i w_M
Weighting $w_i = 1/D$
Step 3
Position determination by weighted averaging reference points
$\hat{\mathbf{x}} = \frac{\sum_{i \in \mathcal{K}(\mathbf{z})} w_i \mathbf{x}_i}{\sum_{i \in \mathcal{K}(\mathbf{z})} w_i}$

Figure 5.2 Illustration of fingerprint positioning using the deterministic method of KNN.

observing certain signal measurements given the user is at $\mathbf{x}_i$). With enough survey samples, one can empirically estimate these likelihoods. At runtime, given a new observation $\mathbf{z}$, the goal is to infer the location. By Bayes' rule, the posterior probability that the user is at location $\mathbf{x}_i$ given observation $\mathbf{z}$ is:

$$P\left(\mathbf{x}_i|\mathbf{Z}_i=\mathbf{z}\right)=\frac{P\left(\mathbf{Z}_i=\mathbf{z}|\mathbf{x}_i\right)P\left(\mathbf{x}_i\right)}{P\left(\mathbf{Z}_i=\mathbf{z}\right)} \tag{5.3}$$

where $P(\mathbf{x}_i)$is the prior probability of being at $\mathbf{x}_i$ (which might be assumed uniform if no prior knowledge is available) and $P(\mathbf{Z}_i = \mathbf{z})$ is the total probability of $\mathbf{z}$ under all locations. Since $P(\mathbf{Z}_i = \mathbf{z})$ is the same for all candidate locations, the location that maximizes the posterior is also the one that maximizes the likelihood $P(\mathbf{Z}_i = \mathbf{z}|\mathbf{x}_i)$ (if priors are equal, meaning an uniform distribution is assumed). Thus, the maximum likelihood (ML) or maximum a posteriori (MAP) estimate is:

$$i^* = \arg\max_{1\le i\le M} P\left(\mathbf{Z}_i=\mathbf{z}|\mathbf{x}_i\right) \tag{5.4}$$

and $\hat{\mathbf{x}} = \mathbf{x}_{i^*}$ as before. This probabilistic formulation underpins many radio fingerprinting algorithms that use statistical models of signal strength. For instance, the Horus system by Youssef and Agrawala utilized a probabilistic approach to account for Wi-Fi RSS variability [7], modeling the distribution of RSS at each reference point with histograms or parametric distributions and computing $P(\mathbf{x}_i|\mathbf{z})$ to make a decision.

The advantage of the probabilistic approach is that it provides a measure of confidence or likelihood for each candidate location, which can be useful to derive uncertainty estimates or to combine with other information (e.g., combining with inertial sensors via a Bayesian filter, although we do not cover filtering in this chapter). One can also derive a probabilistic location estimate as a weighted average of reference point coordinates:

$$\hat{\mathbf{x}}=\sum_{i=1}^{M}P\left(\mathbf{x}_i|\mathbf{Z}_i=\mathbf{z}\right)\cdot\mathbf{x}_i \tag{5.5}$$

which effectively gives an expectation of the user's location under the posterior distribution. However, in practice, this sum often includes many negligible probabilities and a few dominant ones (especially if the fingerprints are well-separated), so a calculation based on the top k candidates is typically sufficient.

Compared to Figure 5.2 (the illustration based on a deterministic model), the illustration of the one using probabilistic method is shown below.

5.1.3 Deterministic Versus Probabilistic Fingerprinting

To illustrate the difference between deterministic and probabilistic fingerprinting, consider a simple scenario. Suppose that the feature vector is

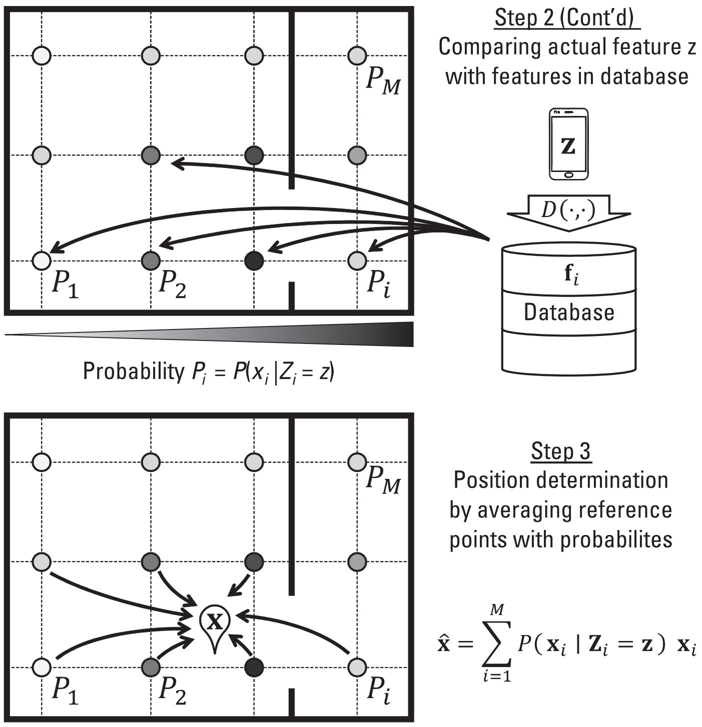

Step 2 (Cont'd)
Comparing actual feature z
with features in database
z
$D(\cdot,\cdot)$
$\mathbf{f}_i$
Database
P_M
P_1
P_2
P_i
Probability $P_i = P(x_i \mid Z_i = z)$
Step 3
Position determination
by averaging reference
points with probabilites
X
$\hat{\mathbf{x}} = \sum_{i=1}^{M} P(\mathbf{x}_i \mid \mathbf{Z}_i = \mathbf{z})\ \mathbf{x}_i$

Figure 5.3 Illustration of fingerprint positioning using the probabilistic method of MAP.

{AU: Call out figure sequentially in text}

one-dimensional ($N = 1$) measuring, say, the RSS from a single Wi-Fi AP. Three reference points A, B, and C have mean RSS values of –40 dBm, –60 dBm, and –80 dBm, respectively, at locations a_A, x_B, x_C. In a deterministic approach, if a user measures $z = -58$ dBm, one might compute distances: $|-58 - (-40)| = 18$, $|-58 - (-60)| = 2$, and $|-58 - (-80)| = 22$ for locations A, B, and C, respectively. The nearest neighbor is B (distance 2), so we pick location x_B. A probabilistic approach would model that each reference point's RSS is a random variable (due to fading and noise). If at B the distribution is centered at –60 with some variance, we could compute the likelihood of –58 at each point: $L_A = p(Z_A = -58|x_A)$, $_{LB} = p(Z_B = -58|x_B)$, and $L_C = p(Z_C = -58|x_C)$. If we assume Gaussian distributions (which is used in many practical applications) for simplicity:

$$L_i = \frac{1}{\sqrt{2\pi}\sigma_i}\exp\left(-\frac{(-58-\mu_i)^2}{2\sigma_i^2}\right) \tag{5.6}$$

where μ_i is the mean RSS at reference i and σ_i is the standard deviation. Even if μ_B is slightly different from the observation, a larger σ_B (wider distribution) could make L_B comparable to L_A if A had a smaller variance but the value is far from μ_A. In any case, the outcome is typically still that x_B has the highest probability (unless −58 was an unlikely reading for B and more likely for A due to overlapping distributions).

Deterministic methods such as *k*-NN are simpler to implement and often perform well if the feature space distances correlate with physical distance. Probabilistic methods can yield better results when properly calibrated because they account for the variability of measurements; for example, far-from-router areas might have very fluctuating signals that a deterministic method could mishandle, whereas a probabilistic method might down-weight those readings due to broad distributions. Some systems in the literature compare these approaches and sometimes hybridize them (e.g., using *k*-NN to get a set of candidates and then applying Bayesian weighting) [8].

5.1.4 Advanced Enhancements in Fingerprinting

Over the years, many enhancements to basic fingerprinting have been proposed. We briefly mention a few notable ones:

- *Signal processing and filtering:* Before matching, raw measurements can be filtered or averaged. For instance, taking multiple RSS scans and using a moving average can reduce random noise. Some approaches also filter out unstable AP signals (heuristically) or use techniques like Kalman filtering on the measurement sequence to exclude those with higher residuals.
- *Dimensionality reduction:* When N (number of features) is large (e.g., dozens of APs), algorithms such as principal component analysis (PCA) can be applied to focus on the most informative linear combinations of signals [9]. This can speed up matching and sometimes improve accuracy by ignoring redundant dimensions. Nonlinear reduction such as autoencoders (in the ML context) have also been used, as discussed in Section 5.2.
- *Clustering and indexing of fingerprints:* To scale the fingerprints to large environments, the database can be clustered or indexed to avoid comparing against all M fingerprints every time. One method is to partition the area into regions (using, e.g., *k*-means clustering on the fingerprint feature space) and only search within the region whose centroid is closest to $\mathbf{z}$. Another is to use tree structures or hashing

for high-dimensional search. These techniques improve computational efficiency, which is important for real-time systems.

- *Continuous space estimation:* Instead of assuming that the user's location must coincide with a surveyed point, various interpolation schemes create a continuous radio map. For example, one can use Gaussian process regression to model the signal strength as a continuous function of location [10]. This yields a predictive mean and variance of the signal at any coordinate, which can then be used in a likelihood formula to find the most likely location via continuous optimization. Such methods blend fingerprinting with radio propagation modeling, offering a trade-off between purely empirical and theoretical approaches. Ideally, these techniques can reduce the effort in building the offline database.
- *Crowdsourcing and automated updates:* A major challenge for fingerprinting is keeping the database fresh (i.e., up-to-date) as environments change. Modern approaches allow users' devices to contribute new fingerprints as they move around (with either opportunistic or explicit reporting of their locations), thereby continually updating the database. The system might use the incoming data to detect changes (e.g., an AP moved or a new AP added) and adapt. Although this enters the realm of localization infrastructure management rather than the core algorithm, it is facilitated by robust fingerprint matching that can handle a degree of variation.

Fingerprinting has proven to be effective in many real-world deployments, achieving typical accuracies on the order of 5 to 10m for Wi-Fi-based systems in office buildings [3]. However, its performance can degrade if the environment or device conditions differ significantly from the training data. This motivates more adaptive and learning-based approaches, which we cover in later sections.

5.2 Pattern Recognition Approaches for Indoor Positioning

The problem of indoor localization via feature matching can be viewed through the lens of pattern recognition. In this view, the goal is to recognize the user's location based on observed patterns (features) and how one would classify an object based on its attributes. Many classical pattern recognition algorithms can be applied to fingerprint data to improve accuracy or reduce computational load, beyond the basic nearest-neighbor matching discussed earlier.

In this section, we discuss how indoor localization can benefit from established pattern recognition techniques. We cover classification approaches that treat location estimation as a discrete classification problem, regression approaches that predict continuous coordinates, and feature engineering techniques that extract more robust patterns from raw sensor data. We also highlight how some methods explicitly leverage domain-specific patterns, such as building layout constraints or motion patterns.

5.2.1 Location Estimation as a Classification Problem

One way to frame indoor positioning is as a multiclass classification problem. Suppose that the area of interest is divided into a finite set of C distinct locations or zones (these could be cells in a grid, specific rooms, specific floors, or simply the set of reference points themselves). Each class $class \in (1, 2, \ldots, Classes\}$ corresponds to a location or region in the building. During the offline phase, we gather feature vectors for each class. In the online phase, given an observed feature vector $\mathbf{z}$, the task is to predict to which class (location) it belongs.

There are standard classification algorithms that can be trained for this task:

- *Decision trees and random forests:* A decision tree classifier can be trained on the fingerprint data with location classes as labels. The tree will learn thresholds on feature values to split the space into regions corresponding to different locations. Decision trees are human-interpretable to some extent (e.g., they might learn rules such as “if AP1 signal > −50 dBm and AP2 signal > −60 dBm, then Location = Lobby”). An ensemble of trees (i.e., random forests) often improves accuracy by reducing overfitting, especially given the high-dimensional and possibly noisy fingerprint data.
- *Support vector machines (SVM):* SVMs seek to find hyperplanes in the feature space that separates classes with maximum margin. For indoor localization, one can train an SVM to distinguish between every pair of location classes. However, SVMs are more naturally binary classifiers; multiclass SVM approaches (one-versus-one or one-versus-rest) can be applied if the number of distinct locations is not too large. In a large building with hundreds of reference points, using SVM directly can be computationally intensive and may require kernel methods to handle class complexity. Nevertheless, for smaller-scale problems or for distinguishing between broader areas, SVMs have been applied.

{AU: Please change this to the specific section number}

- *Bayesian classification:* As introduced in the fingerprinting section, one can take a Bayesian pattern recognition approach. Naive Bayes classifiers, which assume feature independence, have sometimes been used with quantized signal features. More complex Bayesian networks or even discriminative models such as logistic regression (which essentially fits a logistic function to approximate $P(\text{location} = class|\mathbf{z})$) can also be used. These approaches are closely related to the probabilistic fingerprinting discussion; the difference is often in whether the model is generative (model $P(\mathbf{Z}|\mathbf{x})$ as in Naive Bayes) or discriminative (directly model decision boundaries as in logistic regression or SVM).

Each of these classifiers has advantages and disadvantages. Simpler classifiers such as decision trees are easy to implement and can handle large numbers of classes (locations) but might not capture subtle joint patterns in the features. More complex ones such as SVM or Bayesian methods can potentially be more accurate if tuned well, but they might require significant training data and careful parameter selection (e.g., kernel choice for SVM, or distribution assumptions for Bayes). Empirically, studies have shown that these advanced classifiers can improve localization accuracy in challenging scenarios. For example, Brunato and Battiti demonstrated that SVM-based matching yielded better performance than a weighted *k*-NN baseline in a WLAN localization testbed [11]. However, such gains come with increased computational costs and complexity of model training.

One practical consideration when applying classification algorithms is that the number of classes (distinct location labels) can be very high if one treats every reference point as a separate class. If M reference points exist, then $Classes = M$ in that formulation. Some locations might be very similar in their fingerprints, effectively making the classification problem extremely difficult for those classes (they are not linearly separable or even easily nonlinearly separable). To address this, one strategy is to reduce the granularity of classes by grouping nearby reference points into one class or region, meaning clustering. This sacrifices some precision (the system will output a region or cell rather than an exact point), but it can make the classification task tractable. Subsequently, a finer positioning within that region can be done by other means (perhaps a local fingerprint interpolation or an additional sensor cue). This two-step hierarchical approach, as demonstrated in Figure 5.4, is sometimes used in large buildings: first classify the area (which floor, building, or wing or which sector of a floor), and then pinpoint within that area.

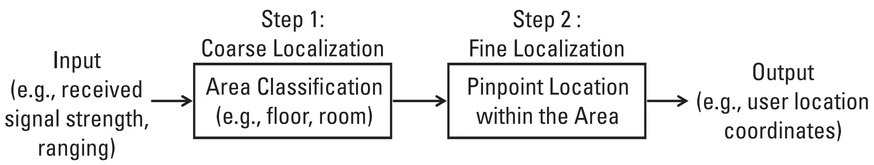

Input
(e.g., received signal strength, ranging)
Step 1:
Coarse Localization
Area Classification
(e.g., floor, room)
Step 2 :
Fine Localization
Pinpoint Location within the Area
Output
(e.g., user location coordinates)

Figure 5.4 The two-step hierarchical approach for indoor positioning.

5.2.2 Regression Approaches for Coordinate Estimation

Instead of classifying the location into a discrete set, one can attempt to learn a direct mapping from features to continuous coordinates (x, y) (and z if height is included). This is a regression problem: given a feature vector $\mathbf{f}$, predict $\mathbf{x} = [x, y]$. Classical regression algorithms can be employed. Here we introduce popular methods:

- *Multivariate linear regression:* In theory, one could fit a linear model $\hat{\mathbf{x}} = W\mathbf{x} + \mathbf{b}$ where W is a $2 \times N$ matrix (for 2-D output) and $\mathbf{b}$ is a bias vector. N is the input feature numbers (e.g., number of APs). Each row of W and the corresponding bias would form a linear combination of the input features to predict x and y. However, indoor propagation is highly nonlinear with respect to position, so a linear model will generally not fit well except perhaps in very small, simple areas. That said, linear regression might be used in a piecewise manner if the environment is partitioned.
- *Gaussian process regression:* As mentioned in the fingerprinting section, Gaussian processes (GP) can model the mapping from location to signal, but, conversely, they can also be used to infer location from signal by inverting the relationship in a probabilistic sense. If we have a prior that the location coordinates are a function of the features, a GP could be trained in the reversed way (although typically it is easier to model signal as function of location, not vice versa, due to dimensionality differences). Instead, one might train two GPs, one for x and one for y, each as a function of the feature vector. However, directly using a high-dimensional feature vector as input to a GP might be impractical for large N.

{AU: Word missing here, or should this be "a priori"?}

- *Support vector regression (SVR):* The regression counterpart of SVM can be used to predict continuous coordinates. SVR attempts to find a function (nonlinear if using a kernel) that fits within an ϵ-tube (which defines the tolerance for errors in the prediction of numerical values) of the training data and is as flat as possible, meaning minimizing complexity. For localization, one could train two SVR models, one for *x* and one for *y* (or use a multi-output SVR if available). This method has been explored in some studies with moderate success [12]; it often requires careful tuning of kernel parameters. SVR tends to be slower than a simple neural net or tree once trained, because it involves computing distances to support vectors. However, one advantage is that SVR inherently performs a form of regularization, which can prevent overfitting.

- *Ensemble methods:* Aside from random forests (which we covered as a pattern recognition classifier), other ensemble approaches such as Gradient Boosting Machines (GBM) or Adaptive Boosting (AdaBoost) have been tried for indoor location classification. AdaBoost can combine many weak classifiers into a stronger one, and some experiments used decision stumps (one-level decision trees) boosted to create a robust location predictor [8]. The ensemble approach can handle the multiclass nature and often yields good accuracy, but, similar to SVM, the computational load might be higher, and the model can become a bit of a black box.

The regression perspective is appealing because it directly aims to minimize location error during training. However, pure regression on high-dimensional, sparse data (the surveyed fingerprints are only known at reference points) can be tricky. Often, a classification step (coarse localization) is followed by regression (fine localization), or regression models are used to refine a classification or nearest-neighbor result. When using these regression methods, the model training is typically done offline with the collected fingerprint dataset. One must be careful to avoid overfitting between the input features and the reference points. A common practice is to evaluate the model on validation data collected at distinct locations or at times different from the training phase to ensure that the model generalizes.

Another interesting direction in pattern recognition is semi-supervised learning or manifold learning. In some cases, one may collect a lot of unlabeled sensor data (no ground truth position) while only some fraction is labeled (with ground truth). Algorithms such as self-organizing maps or manifold alignment might be used to cluster the unlabeled data, potentially revealing the structure of the space [13]. Then a smaller number of labeled points can anchor those clusters to physical locations. This is beyond the scope of our chapter to derive fully, but it is worth noting as a pattern recognition approach, especially when labeled data (ground-truth locations) are expensive to get.

{AU: Do you mean "discuss"?}

5.2.3 Feature Engineering and Dimensionality Reduction

In pattern recognition, how you represent the data (feature engineering) is often as important as the choice of algorithm. For indoor positioning, the raw features might be, for example, a list of RSS values from various APs. However, these raw features can be transformed or expanded to improve performance. Next are popular methods. M denotes the total fingerprint data epoch, and N denotes the number of features in each fingerprint data epoch.

- *Feature selection:* Not all AP signals may be informative. Some may be too weak or sporadic to be useful, or some may be highly correlated with others. A common step is to select a subset of the strongest or most stable AP signals to use in the fingerprint. This reduces M and often improves robustness. Algorithms such a greedy forward selection [14] or backward elimination [15] can be employed, using cross-validation error as a criterion to pick the best subset of features for location prediction.
- *PCA:* As mentioned, PCA can reduce noise. One might take the collected $M \times N$ matrix of fingerprints and compute its principal components and then represent each fingerprint by the first k principal component coefficients. If most of the variance (information) is retained in those k components, then matching or classification can be done in this reduced k-dimensional space. Importantly, PCA is an unsupervised method (it does not use location information in finding components, just the feature correlations), so it aims to capture the structure of the feature data, which might align with spatial differences but not always. Still, it can help when N is very large or if signals are very noisy.
- *Discriminant analysis:* Methods such as linear discriminant analysis (LDA) explicitly try to project data into a lower-dimensional space that maximizes separability between classes (locations). If treating each reference point as a class, LDA could, in principle, yield a projection that emphasizes differences between reference points. However, LDA typically requires more samples than dimensions for stability, which can be an issue if $N > M$ (only few epochs of data are collected) or if there are many classes. In practice, one might use LDA on a higher-level grouping (such as room-level classes) to get features that separate rooms well.

Physical constraints of the environment can also be incorporated into pattern-based localization. For example, after a fingerprint-based estimate is obtained, it can be adjusted to the nearest valid location on the floor plan (e.g., shifting an estimate that falls inside a wall to the closest open area). In this way, pattern matching is augmented with geographic context, improving the realism and feasibility of the position estimate.

Lastly, we note that any pattern recognition approach should ultimately be evaluated on how well it estimates location, not just the classification accuracy. For instance, if a classification approach predicts the wrong room but that room is adjacent to the correct one, the location error might be small if the user was near the boundary. At this time, an error uncertainty metric would reflect a small error, whereas classification accuracy would mark it as a failure. Therefore, the design of the system often involves a combination of classification and regression thinking. Many competitions and benchmarks in indoor localization (such as the annual IPIN competition with different tracks of data corresponding to different user scenarios [16]) use accuracy metrics such as mean error distance, median error, or percentage of estimates within 1-2-5 meters, which guide how algorithms are tuned.

{AU: This is unclear; do you mean "1m, 2m, or 5m"?}

5.3 Deep Learning-Based Approaches

Deep learning techniques have increasingly been applied to indoor positioning, building upon the pattern recognition approaches with more advanced models and training methodologies. In many cases, the line between pattern recognition and machine learning is blurred, as techniques such as decision trees or SVM are indeed part of machine learning. However, in this chapter, we use the term "machine learning-based approaches" to emphasize methods that involve an explicit training phase to learn complex models (often with many parameters) from data, including neural networks and deep learning, which have shown promising results in recent years [17]. Unlike the simpler methods that might use direct matching or relatively straightforward classification boundaries, advanced deep learning models can capture intricate nonlinear relationships in fingerprint data, potentially improving accuracy and robustness. They can also naturally handle multimodal inputs and incorporate contextual information if available.

In this section, we first discuss deep learning approaches, such as recurrent neural network and convolutional neural networks, that have been adapted for fingerprint-based positioning. We also address practical considerations such as training data requirements and model generalization.

5.3.1 Algorithms for Localization

Deep learning refers to neural networks with multiple layers that can automatically learn hierarchical feature representations. A typical neural network architecture might have N input neurons (for the N-dimensional feature vector), one or more hidden layers with nonlinear activation functions (e.g., sigmoid or ReLU), and output neurons corresponding to either coordinates or class labels. In a typical coordinate-regression network, there would be 2 (or 3) output nodes for x, y (or z), and the network is trained to minimize mean squared error between the outputs and the true coordinates. In a classification network, there would be M output nodes (for M location classes or reference points), with a softmax activation and cross-entropy loss, so the network is trained to output a probability distribution over locations given the input. In 2002, Battiti et al. showed that neural networks could achieve comparable or slightly better accuracy than k-NN when trained well [18], although, in the early 2000s, the size of networks and amount of data were limited.

A deep learning approach can combine the advantages of both classification and regression. If a classification approach yields an estimated class (such as a specific reference point identification), one can post-process that into coordinates (by mapping that identification to known coordinates, or using a centroid if the class was a region). If the model yields a continuous coordinate, that is straightforward, but sometimes a hybrid output is used (e.g., an NN that outputs both a classification distribution and a coordinate). Figure 5.5 shows a classic flowchart on how to apply deep learning algorithms to indoor feature-matching localization.

The popular deep learning networks are briefly introduced here.

- *Fully connected neural networks (FCNN):* Also known as multilayer perceptron (MLP) with many hidden layers. These can be seen as an extension of the simple NN approach mentioned above, but with more layers and neurons. In practice, simply stacking more layers on an FCNN for fingerprinting can yield some gains, but there is a risk of overfitting without enormous data. Researchers have used techniques like dropout (randomly dropping units during training) and L_2 regularization to improve generalization. Some deep networks use one-hot encoding of location identification as output with a softmax and then convert to coordinates if needed (for example, by outputting the weighted centroid of the highest-probability classes). This network is classic and usually a good starting point for researchers who just stepped into the field.

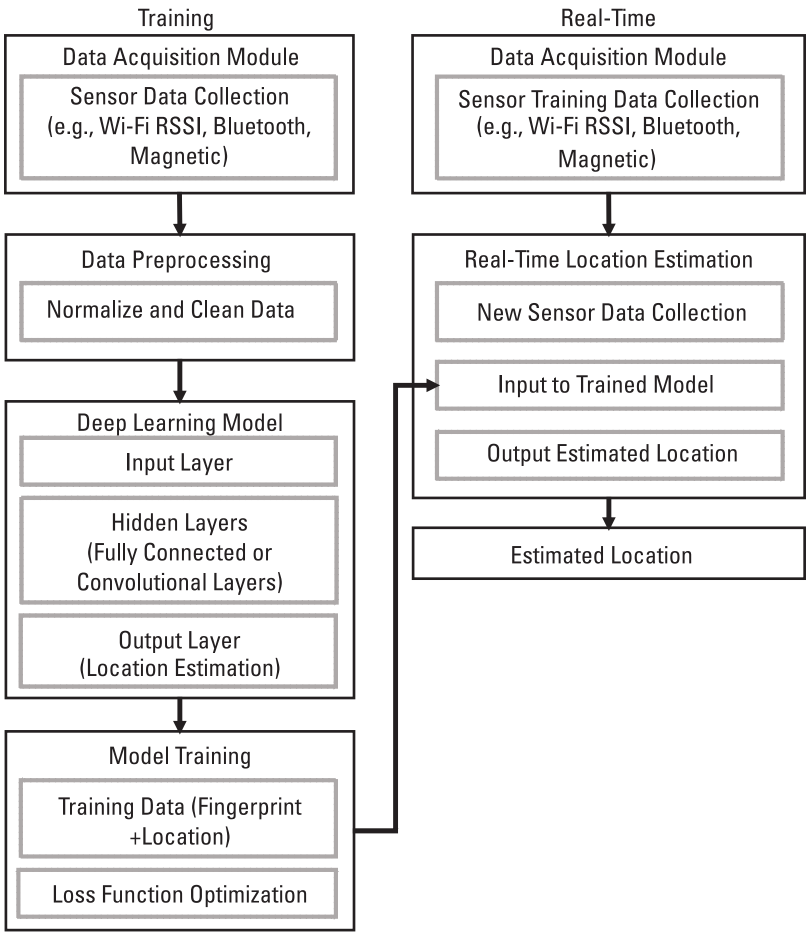
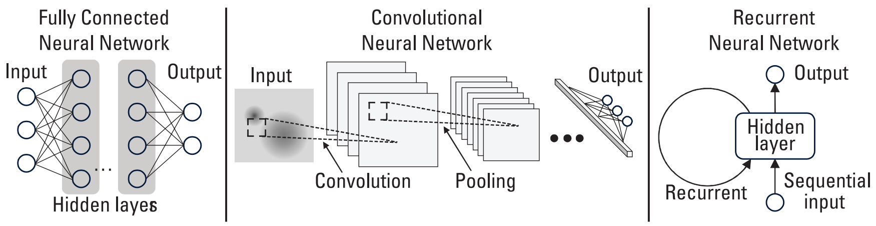

Training
Data Acquisition Module
Sensor Data Collection (e.g., Wi-Fi RSSI, Bluetooth, Magnetic)
Data Preprocessing
Normalize and Clean Data
Deep Learning Model
Input Layer
Hidden Layers (Fully Connected or Convolutional Layers)
Output Layer (Location Estimation)
Model Training
Training Data (Fingerprint +Location)
Loss Function Optimization
Real-Time
Data Acquisition Module
Sensor Training Data Collection (e.g., Wi-Fi RSSI, Bluetooth, Magnetic)
Real-Time Location Estimation
New Sensor Data Collection
Input to Trained Model
Output Estimated Location
Estimated Location

Figure 5.5 An illustration of a deep learning model for fingerprint-based indoor positioning.



{AU: Call out figure sequentially in text}

Figure 5.6 Popular deep learning network architectures [19].

- *Convolutional neural networks (CNN):* CNNs are typically used for grid or image-like data. One innovative approach in Wi-Fi fingerprinting is to treat the list of RSS readings as a sort of 1-D image or as a 2-D pseudo-image. For example, one can arrange the signals from different APs into a 2-D matrix (perhaps sorting APs by their physical location or frequency channel) and treat that as an image in which pixel intensity corresponds to signal strength. Then a CNN can be applied to learn spatial filters on this matrix [20]. Some studies have shown that CNNs slightly outperform FCNN for fingerprint data, possibly because they can capture the spatial correlation of the local patterns (such as a group of APs that often vary together). It is also worth mentioning that, in some cases, framing the fingerprint data as an image can incorporate knowledge of physical layout. One approach is to create a 2-D grid overlay on the floor plan and "paint" the signal strengths onto this grid at the reference point locations, effectively making a sparse image for each AP or each fingerprint. Then one might use a CNN to regress from this image to the user's position. This sort of spatially aware deep learning incorporates knowledge of the building's geometry. However, it requires preprocessing to align data with coordinates.

{AU: Call out reference 19 sequentially in the text}

- *Recurrent neural networks (RNN):* If a user is walking, one can consider the sequence of recent fingerprints. RNNs or long short-term memory (LSTM) networks can take a time series of observations and output an updated location estimate, effectively learning to smooth or track the user. This borders on tracking (which might involve filtering), but purely data-driven sequence models can be trained to incorporate the user's motion patterns or the temporal stability of signals. For example, a user walking down a hallway will have gradually changing Wi-Fi signals; an RNN could learn to interpret a sequence such as [–60, –58, –55,…] from a particular AP as indicative of moving toward that AP, thus adjusting position estimates accordingly. This is an interesting direction where machine learning is used for what traditionally might be handled by a Kalman filter or particle filter.
- *Transformer networks (transformers):* Transformers are a newer deep learning architecture that utilizes self-attention mechanisms to model complex dependencies in data. Unlike RNNs, which process sequences step by step, transformers process the entire input simultaneously, allowing them to capture long-range dependencies more effectively. In the context of indoor localization, transformers have been explored for both Wi-Fi fingerprinting and multisensor fusion. One approach is

to treat each signal source (e.g., an AP or beacon) as a "token" and let the transformer dynamically learn which signals are most relevant to estimating location [21]. This makes it inherently adaptive to varying signal conditions. Another application is using transformers for sequential fingerprint data, where the model processes a time-series of RSS readings and learns temporal patterns for improved localization accuracy. Compared to LSTMs, transformers excel at handling long-term dependencies while being more parallelizable, making them efficient for batch processing. However, standard transformers require significant resources, so lightweight adaptations (e.g., pruning attention heads, quantized models) are necessary for real-time mobile deployment. Recent studies have indicated that transformer-based models can outperform LSTMs and CNNs in complex environments, particularly when exploring the correlation between continuous inputs [21]. Nonetheless, their data hunger and computational overhead remain challenges, making them most suitable for large-scale, high-accuracy applications rather than lightweight, on-device inference.

- *Autoencoders and unsupervised feature learning:* Autoencoders are neural networks trained to compress and reconstruct data. In fingerprinting, an autoencoder can be used to learn a lower-dimensional representation of the high-dimensional feature vector (similar to PCA but potentially nonlinear). One could train an autoencoder on all the fingerprint vectors $\{\mathbf{f}_i\}$ to get, say, a 10-dimensional code that captures most of the information. Then use those 10-dimensional codes as features for a simpler localization algorithm (such as *k*-NN or a small classifier). The benefit is noise reduction and focusing on key variations. Furthermore, if one has a large amount of unlabeled data (fingerprints without known locations), an autoencoder can be trained on that to improve the representation, and then a smaller labeled set is used for supervised learning on top of the learned features (this is a form of unsupervised pre-training).

Deep learning models often achieve high accuracy in the training environment but can suffer if the environment changes (e.g., furniture moved, AP transmit power changed). One solution is to periodically retrain or fine-tune the model with new data. Another approach is to design the model outputs in a way that provides uncertainty information. For example, a Bayesian neural network or an ensemble of networks could give a distribution over position rather than a point estimate, which can be useful in practice to know when the model is confident versus when it is unsure.

Because deep models can be data-hungry, there have been efforts to create large public databases of indoor localization fingerprints. For example, the UJIIndoorLoc-Mag dataset [22] contains over 20,000 Wi-Fi fingerprints collected in a multibuilding campus and has been used to benchmark many algorithms including deep networks. Results from such evaluations show deep learning to be among the top performers, often achieving median errors around 1 to 2m in those scenarios, which is a notable improvement over traditional methods on the same data [17].

Furthermore, recent research has explored few shot and transfer learning techniques to enable models to generalize to new buildings or floors with very little new training data, which is crucial for making deep learning solutions more practical in dynamic real-world deployments. For example, in practice, one may have a trained model in one building and want to use it in another with some adaptation. Transfer learning in deep networks might involve taking a model trained on one large dataset and fine-tuning it on another smaller dataset, rather than starting from scratch. This has been popular in image-based localization (taking networks pre-trained on large image datasets and fine-tuning on specific building images). For radio fingerprinting, transfer learning might involve transferring knowledge of general signal patterns. Some researchers have also looked at simulation-to-real transfer: train a deep network on simulated radio maps (using ray tracing or other propagation models for many virtual environments) and then adapt it to real measurements with minimal real data [23]. This is still an emerging area.

{AU: Correct?}

5.4 Key Challenges and Future Directions

While integrating features matching with machine learning methods have shown great promise for indoor positioning, they also face a number of challenges. In this section, we outline some key challenges and discuss ongoing research directions aimed at addressing them.

5.4.1 Data Scalability and Deployment Effort

A recurring theme with machine learning approaches is the reliance on data and one major challenge with fingerprinting methods is the effort needed to deploy them in a new environment. Collecting a fine-grained fingerprint database is labor-intensive. If a building has many floors and rooms, surveying every area with sufficient density of reference points can be time-consuming. In the early stages of the first author's career, collecting fingerprints for a classroom already took him a half-day as shown in Figure 5.7 [24].

{AU: Edit correct?}

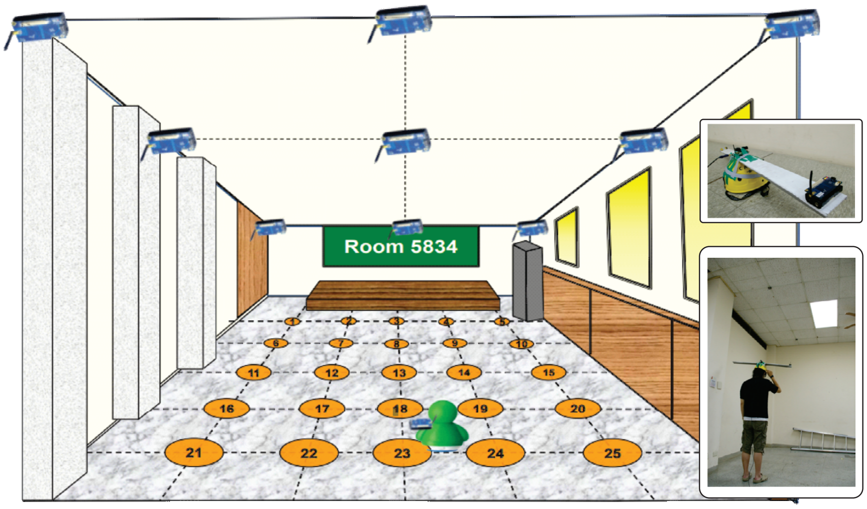


Figure 5.7 An illustration of collecting fingerprint data from 9 ZigBee transceivers at 25 grid points. The data collector stands at each of the grid points for 5 minutes.

The larger the area, the more data storage and computation are required for matching. Techniques such as data augmentation are used to synthetically increase the diversity of training data without additional surveys. For example, one could add random noise to existing fingerprints to simulate measurement noise or perturb a known location's fingerprint by interpolating with a neighbor's to simulate an in-between location. In vision, data augmentation (rotating, scaling images) is common; for RSS fingerprints, adding small Gaussian noise or dropping out a random AP reading (to simulate a temporary signal loss) could help the model to be robust to those events [25].

To make things harder, a variety of devices might measure signals differently (due to calibration offsets, different antenna gains). If the training data is collected with one device and a user uses another, the performance can degrade. Approaches to handle this include calibrating the devices (if possible) or including data from multiple device types in training, but this is not too cost-effective. Another approach is to use relative features such as the rank of signals or to normalize the feature vector (for example, subtract the mean and divide by the standard deviation of RSS values for that scan) so that differences in overall sensitivity are mitigated. However, this normalization method does not always work if the two devices generate measurements with very different probabilistic behaviors.

In addition, indoor environments are not static. Furniture can be moved, doors opened or closed, people come and go (causing signal shadowing and changing multipath), and electronic devices might be added or removed (introducing new signals or interference). A fingerprint collected at one time may gradually become outdated. This concept is known as temporal degradation of fingerprints. Studies have shown that Wi-Fi fingerprints can remain usable over months, but accuracy might slowly worsen as the environment changes [26]. For magnetic fingerprints, large changes are less frequent (the building structure would have to change), but small changes such as new electronic equipment can perturb readings. Periodic recalibration could be conducted to update the database. For instance, update readings in areas that users frequent the most. More recently, approaches such as crowdsourcing (users contribute data) have been proposed to reduce the burden on a dedicated surveyor [27], but then ensuring the quality and proper labeling of crowd-collected data becomes an issue. These are discussed in Chapter 9.

Even if we frequently update the database to refine the model, users and surrounding dynamics can affect the real-time reading of the user device. The robust function mentioned in Chapter 2 can also alleviate this. Another good idea is to increase the data heterogeneity (e.g., using multiple technologies including Wi-Fi, cellular, magnetic) to increase the chance that some signals remain stable even if others fluctuate.

Other than the problems of data, the computational cost could also be an issue. Training a deep neural network can be computationally intensive (see Figure 5.8). Evaluating a neural network (especially if small to medium-sized) on a mobile device can be feasible due to optimized libraries and the relatively low-dimensional input. Some systems offload computation to a server (cloud localization services) to run heavy algorithms and send results back to the device. The choice may depend on privacy (sending raw sensor data out) versus device capability and latency requirements. In summary, machine learning has brought powerful tools to indoor positioning, enabling high accuracy in

{AU: Correct as added?}

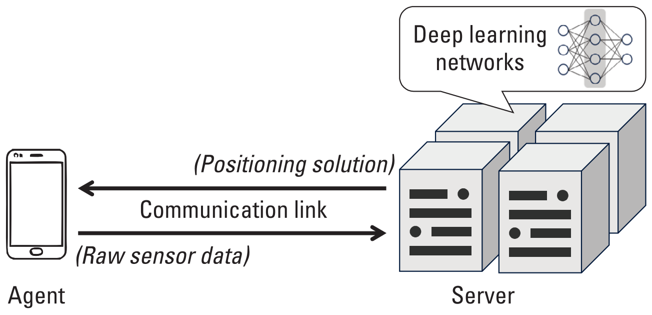


Figure 5.8 An illustration of conducting computationally intensive positioning algorithms by a centralized server with a communication link.

fingerprint-based systems. However, the success of these approaches is tightly coupled with data quality.

5.4.2 Context Awareness and Semantic Localization

Another direction is to incorporate context or semantic information. If the system knows, for example, that the user is on the second floor (maybe from an elevator sensor or user input in an app), it can eliminate matches to first-floor fingerprints entirely, improving accuracy. Context can come from user behavior (if the user is known to be in a meeting scheduled in a certain room, the system might give higher probability to that room). These ideas go beyond pure signal matching and into the realm of context-aware computing. They highlight that feature-matching algorithms can be part of a larger intelligent system that uses multiple sources of information.

For example, a sudden change in barometric pressure reading combined with a distinctive Wi-Fi signature might indicate that a user moved from one floor to another. Similarly, detecting a pattern of several consecutive door openings could hint that the user is moving through a corridor or checking rooms. By fusing such contextual cues with traditional fingerprints, the system can refine its estimates and even provide meaningful place labels (e.g., "in the elevator" or "in conference room 2") rather than just coordinates. This means that the semantic can be used for localization and that is actually how we human being navigate ourselves intrinsically. Pattern recognition can be applied to sensor data to recognize such semantic places, and machine learning can fuse different sensors to improve place recognition. For example, an algorithm might learn that a certain pattern of walking (detected by accelerometer) followed by a sharp drop in light level (entering a dark room) plus certain Wi-Fi signals corresponds to entering a specific lab room. This is quite complex to engineer manually, but deep learning could potentially learn these correlations if given enough training data with semantic labels.

5.4.3 Unavailability of Fingerprints: Physical Model-Based

Due to the cost and labor, the surveying of fingerprints may not be feasible for the database development. An alternative approach to build such a database is via physical model predictions, for example, simulating signal propagation paths for fingerprints on each location based on a 3-D environmental model. This method relies on prior knowledge of surrounding buildings, walls, and obstacles, as well as an appropriate physical model of the measurement features for realistic fingerprints that capable of determining user's location.

This approach will be demonstrated via an example of indoor positioning by radio signals.

In an indoor environment, radio signal propagation can be modeled (simulated) by ray-tracing algorithms. Ray-tracing algorithms compute the possible reflection and diffraction paths that a signal could take before reaching the receiver. This approach follows the law of reflection, where the angle of incidence equals the angle of reflection. Using a known 3-D map of the environment, ray-tracing can: (1) identify NLOS conditions by determining whether a direct path exists, (2) predict reflection paths that could lead to multipath errors, and (3) correct TDOA and AOA errors by incorporating known reflection-induced delays and angles. With the modeled ray traces, the multipath and NLOS can be detected. Furthermore, they can be regarded as additional features for indoor positioning. This is because the reflections and deflections are usually distinctive spatially, meaning that the multipath effects are very different if the location of the agent is different.

To utilize the modeled ray traces, the position-hypothesized method, such as a particle filter, is popularly developed. This can be illustrated by Figure 5.9. First, several hypothesized states $\mathbf{x}_{p,a}$ of the agent a are distributed based

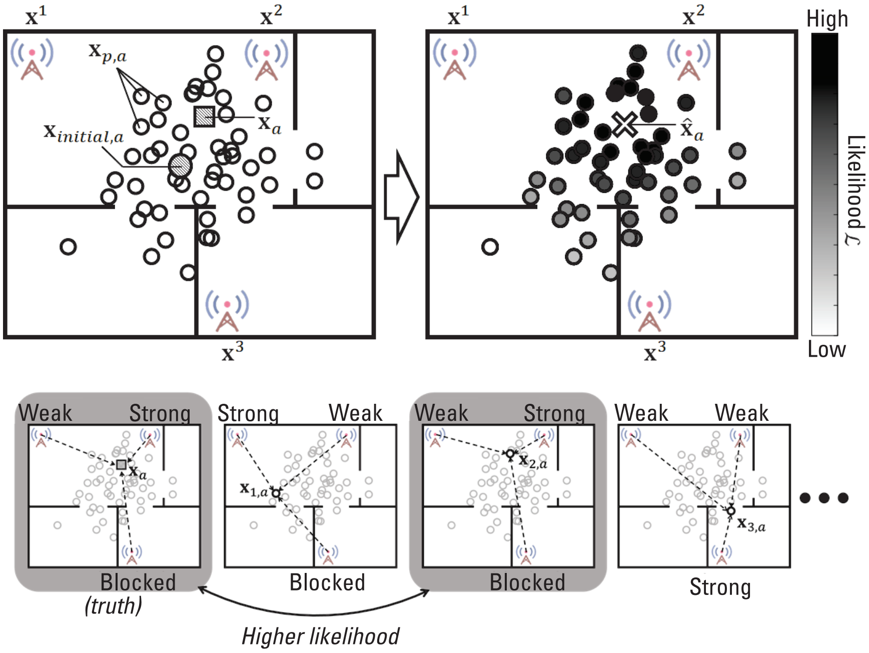


Figure 5.9 Illustration of a position-hypothesized method using a ray-tracing method.

on an initial guess ($\mathbf{x}_{\text{initial}}$, which can be provided by the LS methods mentioned earlier in Chapter 4 or commercial solutions). They can be distributed as grid points or as Gaussian random variables. The key is the uncertainty of the initial guess that has to be considered for distributing the hypothesized states, which is as follows:

$$\mathbf{x}_{p,a} \sim \mathcal{N}\left(\mathbf{x}_{\text{initial},a}, \sigma_{\text{initial},a}{}^{2}\right) \tag{5.7}$$

where $\sigma_{\text{initial},a}$ denotes the standard deviation of the error in the initial guess $\mathbf{x}_{\text{initial},a}$. Then, if one assumes that each measurement z_a^i in $\mathbf{z}_a$ follows independent Gaussian distribution, $z_a^i \sim \mathcal{N}(h_{\text{raytrace}}(\mathbf{x}_a, \mathbf{x}^i), \sigma_{\text{meas\_noise}}{}^2$, where $h_{\text{raytrace}}(\mathbf{x}_a, \mathbf{x}^i)$ denotes the simulated measurement from $\mathbf{x}^i$ to $\mathbf{x}_a$ by the ray-tracing model. The probability density function of each measurement can be represented by

{AU: Edits correct?}

$$f\left(z_a^i \middle| \mathbf{x}_a, \mathbf{x}^i, \sigma_{\text{meas\_noise}}\right) = \frac{1}{\sqrt{2\pi\sigma_{\text{meas\_noise}}{}^2}} e^{-\frac{1}{2}\left(\frac{z_a^i - h_{\text{raytrace}}\left(\mathbf{x}_a, \mathbf{x}^i\right)}{\sigma_{\text{meas\_noise}}}\right)^2} \tag{5.8}$$

For I measurements, the joint probability density function can be written as

$$\begin{aligned} f\left(z_a^1, \ldots, z_a^I \middle| \mathbf{x}_a, \mathbf{x}^1 \ldots, \mathbf{x}^I, \sigma_{\text{meas\_noise}}\right) &= \prod_{i=1}^{I} f\left(z_a^i \middle| \mathbf{x}_a, \mathbf{x}^i, \sigma_{\text{meas\_noise}}\right) \\ &= \left(\frac{1}{2\pi\sigma_{\text{meas\_noise}}{}^2}\right)^{I/2} e^{\left(-\frac{\sum_{i=1}^{I}\left(z_a^i - h_{\text{raytrace}}\left(\mathbf{x}_a, \mathbf{x}^i\right)\right)^2}{2\sigma_{\text{meas\_noise}}{}^2}\right)} \end{aligned} \tag{5.9}$$

Therefore, the likelihood of each hypothesized state can be represented by

$$\begin{aligned} &\mathcal{L}\left(\mathbf{x}_{p,a} \middle| \mathbf{z}_a, \mathbf{x}^1 \ldots, \mathbf{x}^I, \sigma_{\text{meas\_noise}}\right) \\ &= \left(\frac{1}{2\pi\sigma_{\text{meas\_noise}}{}^2}\right)^{I/2} e^{\left(-\frac{\sum_{i=1}^{I}\left(z_a^i - h_{\text{raytrace}}\left(\mathbf{x}_{p,a}, \mathbf{x}^i\right)\right)^2}{2\sigma_{\text{meas\_noise}}{}^2}\right)} \end{aligned} \tag{5.10}$$

The ray-tracing model can be used modeling the TOA, TDOA, RSS, and AOA. There are many commercial solutions on ray-tracing algorithms that can directly be used in (4.103). Finally, by considering all the likelihoods,

{AU: Is this the correct equation?}

the agent’s state can be estimated using a weighted average calculation, where the weighting of each hypothesized state $\mathbf{x}_{p,a}$ is described by its likelihood:

$$\hat{\mathbf{x}}_a = \frac{\sum_p \mathbf{x}_{p,a} \cdot \mathcal{L}\left(\mathbf{x}_{p,a} \middle| \mathbf{z}_a, \mathbf{x}^1 \ldots, \mathbf{x}^I, \sigma_{\text{meas\_noise}}\right)}{\sum_p \mathcal{L}\left(\mathbf{x}_{p,a} \middle| \mathbf{z}_a, \mathbf{x}^1 \ldots, \mathbf{x}^I, \sigma_{\text{meas\_noise}}\right)} \tag{5.11}$$

To implement this method, there are several challenges. The better handling of the following challenges we do, the better positioning performance we can obtain.

1. *The comprehensiveness of the 3-D model:* 3-D models can be described in several levels of detail [28]. For example, if the ray-tracing method cannot have the material of different plane (or even pixel) of the model, the attenuated signal strength of the traced signal cannot be too practical. This is only one attribute. If the shape of the obstacles cannot be well-described by the 3-D models, the accuracy of the ray-tracing model will be affected. Last but not least, the dynamic objects surrounding the agent may also cause reflection effects to the signal received by the agent, resulting in the degraded performance of the likelihood estimation in (5.10).
2. *The complexity of the ray-tracing model:* The ray-tracing method has to consider all the potential blockages, reflections, and diffraction, meaning that it has to consider all the edges and planes within the 3-D models. If multiple reflections during the transmission from a beacon to an agent are considered, the computational load will increase exponentially. As a result, the balance between the computational load to the accuracy of the developed ray-tracing model is also a challenge.
3. *The multimodel issue:* This position-hypothesized method is relying on the assumption that the reflection effects are distinctive between different spatially distributed hypothesized states. However, if the reflections received at two different locations are similar in a way, it may cause a multimodel issue, meaning that there two very high likelihoods at two different locations. It will cause a problem when calculating the weighted average of the hypothesized states, that is, (3.74). This multimodel will frequently occur if the received measurements are very noisy due to the likelihood of different hypothesized states becoming similar. An approach to alleviate the issue is to analyze the likelihoods obtained. In general, the likelihoods should descend with a gradient instead of being uniform-distributed.

{AU: Edits correct?}

{AU: Is this equation correct?}

{AU: Edits correct?}

5.5 Conclusions

In this chapter, we explored indoor positioning techniques based on feature matching, encompassing fingerprinting, pattern recognition, and machine learning approaches. These methods provide an alternative to geometric and signal propagation-based techniques (such as TOA, TDOA, and AOA methods discussed in earlier chapters) by leveraging the unique signatures of the environment. In comparison to traditional methods, fingerprinting and learning-based approaches often excel in environments where obtaining geometric measurements (distances or angles) is impractical or where multipath effects are too complex for simple models. The feature-matching method effectively treats the environment as a black box and learns its behavior empirically. This empirical approach can surpass geometric model-based methods in complex settings, but it requires care to ensure that the learned models remain valid and adaptive to changes. However, these black boxes approach can be an issue when troubleshooting why the system gives a grossly incorrect estimate; it might be hard to reason whether it was due to an overfitted model or a rare pattern that confused it. This actually hinders the continuous advancement of deep learning-based indoor feature-matching system.

While feature matching-based indoor positioning algorithms leverage environmental signatures such as Wi-Fi, BLE, and magnetic field anomalies, these methods inherently depend on the stability of external signals. However, in many scenarios, such as underground facilities, complex indoor areas, or environments with limited infrastructure, external signals may be weak, unreliable, or completely unavailable. Another drawback of feature matching-based methods is that they are environment-specific and inefficient to adapt to new settings; a new environment still requires collecting fresh data and fine-tuning the offline-trained model. To address these limitations, proprioceptive sensors, including inertial measurement units (IMUs) and odometers, offer an alternative approach for positioning based on a user's own motion. Chapter 6 will continue to introduce the use of proprioceptive sensors for indoor positioning.

References

[1] Ou, C. W., et al., "A ZigBee Position Technique for Indoor Localization Based on Proximity Learning," *2017 IEEE International Conference on Mechatronics and Automation (ICMA)*, August 6–9, 2017, pp. 875–880.

[2] Liu, H., et al., "Survey of Wireless Indoor Positioning Techniques and Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 37, No. 6, 2007, pp. 1067–1080.

[3] Bahl, P., and V. N. Padmanabhan, "RADAR: An In-Building RF-Based User Location and Tracking System," *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, Vol. 2, March 26–30, 2000, pp. 775–784.

[4] Haverinen, J., and A. Kemppainen, "Global Indoor Self-Localization Based on the Ambient Magnetic Field," *Robotics and Autonomous Systems*, Vol. 57, No. 10, 2009, pp. 1028–1035.

[5] Wu, K., et al., "CSI-Based Indoor Localization," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 24, No. 7, 2013, pp. 1300–1309.

[6] Li, B., et al., "How Feasible Is the Use of Magnetic Field Alone for Indoor Positioning?" *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, November 13–15, 2012, pp. 1–9.

[7] Youssef, M., and A. Agrawala, "The Horus Location Determination System," *Wireless Networks*, Vol. 14, No. 3, 2008, pp. 357–374.

[8] Dawes, B., and K. -W. Chin, "A Comparison of Deterministic and Probabilistic Methods for Indoor Localization," *Journal of Systems and Software*, Vol. 84, No. 3, 2011, pp. 442–451.

[9] Xia, S., et al., "Indoor Fingerprint Positioning Based on Wi-Fi: An Overview," *ISPRS International Journal of Geo-Information*, Vol. 6, No. 5, 2017, p. 135, https://www.mdpi.com/2220-9964/6/5/135.

[10] Hähnel, B. F. D., and D. Fox, "Gaussian Processes for Signal Strength-Based Location Estimation," *Proceeding of Robotics: Science and Systems*, 2006.

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

[12] Jondhale, S. R., et al., "Support Vector Regression for Mobile Target Localization in Indoor Environments," *Sensors*, Vol. 22, No. 1, 2022, p. 358, https://www.mdpi.com/1424-8220/22/1/358.

[13] Braga, P. H. M., and H. F. Bassani, "A Semi-Supervised Self-Organizing Map for Clustering and Classification," *2018 International Joint Conference on Neural Networks (IJCNN)*, July 8–13, 2018, pp. 1–8.

[14] Deng, K., *Omega: On-Line Memory-Based General Purpose System Classifier*, Carnegie Mellon University, 1999.

[15] Wikipedia, "Stepwise Regression," 2023, https://en.wikipedia.org/wiki/Stepwise_regression.

[16] Potortì, F., "The IPIN Competition 2024," *IEEE Dataport*, 2024.

[17] Alhomayani, F., and M. H. Mahoor, "Deep Learning Methods for Fingerprint-Based Indoor Positioning: A Review," *Journal of Location Based Services*, Vol. 14, No. 3, 2020, pp. 129–200.

{AU: Provide complete publication information}

[18] Battiti, R., N. T. Le, and A. Villani, "Location-Aware Computing: A Neural Network Model for Determining Location in Wireless LANs,", 2002.

[19] LeCun, Y., Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, Vol. 521, No. 7553, 2015, pp. 436–444.

[20] Chen, H., et al., "ConFi: Convolutional Neural Networks Based Indoor Wi-Fi Localization Using Channel State Information," *IEEE Access*, Vol. 5, 2017, pp. 18066–18074.

[21] Zhang, Z., et al., "TIPS: Transformer Based Indoor Positioning System Using Both CSI and DoA of WiFi Signal," *IEEE Access*, Vol. 10, 2022, pp. 111363–111376.

[22] Torres-Sospedra, J., et al., "UJIIndoorLoc-Mag: A New Database for Magnetic Field-Based Localization Problems," *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, October 13–16, 2015, pp. 1–10.

[23] Zhang, C., et al., "Federated Radio Frequency Fingerprinting with Model Transfer and Adaptation," *IEEE INFOCOM 2023—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, May 2023, pp. 1–6.

[24] Jan, S. -S., L. -T. Hsu, and W. -M. Tsai, "Development of an Indoor Location Based Service Test Bed and Geographic Information System with a Wireless Sensor Network," *Sensors*, Vol. 10, No. 4, 2010, pp. 2957–2974, https://www.mdpi.com/1424-8220/10/4/2957.

[25] Amores, E. M. L., "Data Augmentation Models for Improved Indoor Positioning Accuracy Using RSS Fingerprinting," Master's Degree in Data Science, Geographic Information Systems, Satellite Images and Geopositioning, Universitat Oberta de Catalunya, 2024, http://hdl.handle.net/10609/152082.

[26] Mendoza-Silva, G. M., et al., "Long-Term WiFi Fingerprinting Dataset for Research on Robust Indoor Positioning," *Data*, Vol. 3, No. 1, 2018, p. 3, https://www.mdpi.com/2306-5729/3/1/3.

[27] Rai, A., et al., "Zee: Zero-Effort Crowdsourcing for Indoor Localization," *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, Istanbul, Turkey, 2012.

[28] Gröger, G., and L. Plümer, "CityGML—Interoperable Semantic 3D City Models," *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 71, 2012, pp. 12–33.