A Study of Localization based on Quantized Received Signal Strength

by

Alan Yong

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

We bridge the localization techniques that are based on proximity (near/far) information with those based on Received Signal Strength Indication (RSSI) information. An RSSI-based scheme can be mapped to proximity-based scheme by quantizing the RSSI values such they are represented by a single bit. Extending the quantization to multiple bits per RSSI value, we observe diminishing improvement of localization accuracy with more bits used. Quantization of RSSI also enables the offline construction of lookup tables mapping quantized RSSI to coordinates, enabling applications such as: (a) passive localization performed by devices performing table lookups without transmitting, and (b) scalable simultaneous localization of many devices without performing per-device computation. We treat the underlying localization algorithm as a black box and we develop a search process to determine the optimal quantization given a training data set. For our evaluation we use a data set of RSSI measurements in an environment where a vehicle was placed at various positions (or was absent) in order to capture a dynamically changing environment. We evaluate how training and optimizing the quantization on a given data set fares when tested in a modified environment. Our black box localization schemes are taken from the family of k–Nearest Neighbor (k-NN) schemes. Despite their similar results in terms of average accuracy, different members of the k-NN family demonstrate drastically different coverage behavior, i.e., ability for the quantized RSSI values to represent a larger set of distinct locations in space.

Acknowledgements

First and foremost, I give my deepest thanks to my graduate supervisors, Doctors Ioanis Nikolaidis and Janelle Harms. Their extensive experience and insight were crucial in shaping every aspect of thesis and guiding me on fruitful avenues of research. I am extremely grateful for all the time spent on improving and proofreading the text. Additionally, I am grateful for their gentle encouragement and friendly conversations over the years. The work presented in this thesis was partly supported through funding my supervisors received from NSERC's Discovery Grants program.

Next, I would like to express my gratitude to Dr. Paul Lu of the examining committee, not only for the time taken in reading this thesis, but his insightful commentary into future directions this research may take.

To the denizens of the Networks Lab at the University of Alberta, I'd like to say thank you for the enlightening conversations, for being exceptional sounding boards, and for generally being an uplifting presence in my time here.

I extend my heartfelt thanks to my friends, for their unwavering moral support and encouragement through the entirety of my academic studies. Their belief in me kept me going through thick and thin.

Last but not least, I must thank my parents, for their limitless patience and understanding. I would not be here without their constant support and guidance in all aspects of my life. Thank you for always being there for me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One of the success stories of recent decades has been the introduction, and extensive use, of the Global Positioning System (GPS) as the ubiquitous system to determine one's location. Yet, GPS is not helpful in terms of accuracy, or, even, availability, in indoor environments despite the fact that many activities requiring localization are held indoors. The unreliable nature of GPS-based location in indoor environments is rooted in the poor (or completely absent) indoor reception of GPS satellite transmissions. On the other hand, indoor spaces are often awash with wireless transmissions, in particular from wireless access points (APs), wireless Internet-of-Things (IoT) devices etc. A common thread in indoor localization research, followed also in the present thesis, is to perform localization by exploiting transmissions of pre-existing indoor APs/devices.

1.1 Passive GNSS-Denied Localization

Indoor environments are not the only setting where a localization alternative to GPS is needed. An important, possibly alarming, facet of GPS, as well as any other Global Navigation Satellite System (GNSS), is that they represent high value assets and hence candidates for various attacks. Due to the potential devastating impact, attackers can expend significant resources to jam GPS receivers, as well as interfere with, or, even, destroy the operation of GNSS satellites. Hence, being able to operate localization outside of GNSS

infrastructure is a form of resilience under GNSS-denial threats, leading us to the first motivating observation:

Observation 1: Localization without reliance on GNSS adds resilience to location-dependent services.

One valuable lesson from the use of GPS is that it enabled devices to localize *passively*, i.e., just by listening to satellite-transmitted signals, and more importantly, *without* generating transmissions of their own. When looking for alternatives to GNSS, we need to consider as an important property, the ability to perform passive, autonomous, localization. The immediate benefit of passive localization is that the device that is performing localization does not need to expend energy in transmitting for the purpose of localization. As well, by not transmitting, it evades detection insofar Radio Frequency (RF) activity detection is concerned.

Observation 2: Autonomous, passive, localization offers energy efficiency and undetectability benefits.

1.2 Localization Using Limited Resources

GNSS-based localization benefited from the design of special purpose hardware, namely, chipsets custom made for GNSS reception and processing. The specialized chips include very precise RF receiver characteristics coupled with extensive signal processing computation. Effectively, GNSS-based localization requires a hardware investment (the addition of GNSS chipsets) which in recent years has become entrenched and ubiquitous, e.g., in smartphones. Expecting the addition of new specialized hardware to support a second, experimental and evolving, form of localization is an unlikely scenario. Hence, we are led to the following observation:

Observation 3: Non-GNSS localization, is expected to utilize hardware resources already present in devices for other, productive, purposes.

Proposals for non-GNSS localization strategies assume that the devices possess adequate resources (computational, storage, and possibly other hardware resources) to promptly perform localization. The question of whether such resources exist in embedded devices and/or whether the localization can occur in real-time are usually left outside the scope of research studies. This thesis addresses questions related to such embedded implementations of localization.

We will consider solutions involving *computation vs. storage tradeoff*. That is, can we trade storage, e.g., in the form of lookup tables, to reduce the amount of processing necessary for localization? Ideally, the localization described in this thesis is one that can be reduced to a single table lookup, in a potentially very large table, but storage is plentiful and inexpensive. Given that storage is a commodity and invariably exists already in embedded platforms, we argue that it is not a demand for any "special" hardware. Moreover, the generation of those tables will be performed using an upfront one-time offline computation, followed by a one-time imprinting of those tables on each embedded device as part of their deployment.

A side-effect of exploring how to perform passive localization by trading space for computation is that it involves the broader question of runtime complexity of localization. Despite the deluge of papers on localization, their emphasis is placed on how a, *single*, device can determine its location. To the best of our knowledge there is no work on how myriads of nodes can be simultaneously localized in real-time. The literature accepts unquestionably that when many devices are involved, each can independently perform its localization, and to do so promptly.

By accepting the independent localization assumption, the literature indirectly accepts that (a) we are not interested in the *aggregate* computational ef-

fort (and energy) expended across all devices for localization, or, (b), we anticipate that, if the total computational burden is unsuitable for real-time localization on some of the those devices (that are usually resource-constrained), they have to recruit some other devices in performing their localization. Note that (a) implies that even devices that localize to points close to each other, ideally because they are indeed physically near each other, they independently compute a similar result, i.e., in principle wastefully repeating almost the same computation (judging the "sameness" as producing the same result). As for (b), recruiting resources external to the node defeats the purpose of passive localization – the node ends up initiating communication with other, more capable nodes. By the same logic, if localization is performed at scale, extensive aggregate computational resources are expected from the "helper" nodes to serve the under-resourced nodes. The particular mode of operation whereby helper ("server") nodes assist the less resourced ones, is a typical use case of *edge computing* and comes with its own opportunities and complications, but, importantly, if edge nodes need to be recruited, the property of passive autonomous localization is lost. We will attempt to avoid the need for edge computing during the on-line localization phase, to retain the passive localization requirement. In summary:

Observation 4: To achieve passive, scalable, energy-efficient, instantaneous localization in resource-limited devices, a promising approach is to utilize pre-computed lookup tables.

1.3 Quantized RSSI-Based Localization

One can reasonably suspect that any pre-computed lookup tables which are claimed to provide the localization results via a simple lookup operation are either unrealistically large or they may be inaccurate. Such concerns however should be seen under the light of what is the expected accuracy of localization given the particular kind of measurements the devices will use as input to

their localization. As it turns out, and as discussed at length in the main body of the thesis, the measurements we consider are very noisy and convey only a few bits of information as far as their impact on localization is concerned.

Observe that, so far, the motivation behind this thesis has been a scalability argument: reduce computation on devices and localize simultaneously myriads of nodes, e.g., numerous pieces of equipment in a large warehouse. That is, the problem description is indifferent to the particular localization algorithm used, as long as it is an algorithm which depends on receiving transmissions from a number of Radio Frequency (RF) emitters in a, usually indoor, setting. We now narrow the scope, leaving the potential for some strategies laid out here to be applicable to a wider setting of localization algorithms and emitter scenarios, to be revisited in the concluding chapter of the thesis.

For now, let us assume that the emitters whose signals we receive are static and at known locations (in the literature called "anchor," "peg," or "reference" nodes), and we will rely somehow on their known locations to localize unknown points. A companion assumption which can be relaxed as part of the localization algorithms outlined next, is that the anchor emissions are, for the lack of any specific antenna radiation pattern, assumed to be uniform unidirectional. Here we also sneak in an assumption that the localization considers only 2-dimensional space. As long as the unidirectionality on the plane is an acceptable approximation, we do not revisit the issue of emission patterns. While we do not consider the 3-dimensional case, we also do not take any steps that would disallow the generalization of our work to 3-dimensional space. More comments about 3-dimensional extensions are provided in the conclusions section.

An important decision is the choice of emission-related measurement to be used, if no special hardware can be employed. The dilemma here is as follows: recent work has relied on collecting Channel State Information (CSI) but CSI measurements are not widely accessible from commodity transceivers and unlikely to be readily available in the future. CSI-based work reported in the

literature [4, 26, 27] relied on a happy coincidence; a coincidence that allowed the collection of CSI from two commodity IEEE 802.11 (WiFi) device families (Intel 5300 and Atheros 9390), both of which are now obsolete and likely to become unavailable in the near future. There is limited interest from manufacturers to make CSI information available for user-accessible programmatic manipulation. This is because CSI data acquisition requires more resources than the strictly needed resources for the product's primary function – that of being a WiFi transceiver. Failing to collect CSI, the only reasonable alternative available almost universally and exposed via programmatic interfaces is the Received Signal Strenth Indicator (RSSI). RSSI is a scalar, typically expressed in units of dBm, representing the approximate signal strength of the current (or most recent) transmission received by a transceiver. Since the signal strength is impacted by distance, so is the RSSI.

RSSI measurements are ubiquitous, that is regardless of the transceiver technology used. They are also notoriously noisy and have been criticized as being unreliable for localization purposes [12]. Yet, RSSI measurements **do** convey **some** information of value to localization, as several research results have suggested. The approach taken in this thesis is that we reduce the RSSI measurements to a few bits (a single bit in the most extreme case). This transformation is to be performed through a quantization process which splits the range of RSSI values into a few bins, each representing a range of RSSI values. However, as we will later describe, the quantization is informed by how it would impact localization results, allowing us to define a search process to determine the *optimal quantization* for a given set of measurements and an accompanying localization algorithm. Effectively, by representing the bin in which an RSSI value belongs, rather than the entire RSSI value, it is hoped that, coarser, but sufficient information is preserved out of the RSSI values, to achieve good localization.

Observation 5: Access to RSSI measurements is ubiquitous across transceivers technologies, and given their noisy nature, quantizing them is likely to preserve information useful to localization.

1.4 The Family of k–NN Localization Schemes

Notice that we do not elaborate on identifying the source of the noise on RSSI measurements. In doing so, we accept that the measurements are influenced by a multitude of factors, including the impact of the relative orientation of antennae, the quality variations in the manufacturing of the nodes, the differences due to placement with respect to the environment (which could be dynamically changing) etc. Rather than identifying, isolating and compensating for each source of "noise" in the measurements, we treat the measurements as unreliable and only providing a rough indication based on the quantization. However, the choice of how to quantize is not arbitrary – it is a quantization that optimizes a performance metric that captures the accuracy of the localization using quantized values.

Given the plethora of localization schemes, we approach the localization scheme as a "black box". That is, we do not care about the internals of the localization algorithm as long as it operates on the basis that, when given a vector of RSSI values (ostensibly quantized) it produces a 2-dimensional coordinate. It is through this use that, as we will describe in this thesis, the localization scheme is invoked as a subroutine for the process of optimizing the quantization. Once the optimal quantization intervals have been established, we can generate a lookup table as promised and install it on all the devices who wish to passively perform localization.

For the sake of demonstration, we considered the k–Nearest Neighbor (k–NN) variety of localization algorithms in our experiments, although others could have been used as well. The "neighbor" in k–NN is defined over signal space, i.e., over the RSSI measurement vectors at known locations

(the "fingerprint" database collected in a prior survey step). The reasons for choosing k–NN are twofold. First, the k–NN schemes exhibit variety, i.e., there is no single k–NN localization scheme, but rather a family of similar schemes [2, 9, 10, 21] (detailed in the related work section), differing in the way they combine the coordinates corresponding to the k found neighboring vectors. Different members of this family exhibit different degree of performance in response to the RSSI quantization. A second reason is that k–NN is still a computationally non-trivial task (details are provided in Section 2.1.7), hence a good candidate to replace it, for the sake of scalability, with simple table lookup.

Observation 6: The family of k–NN localization schemes is a diverse family, expected to exhibit variety of performance behavior under quantized RSSI, and it stands to scale from RSSI quantization by replacing linear time computation with lookup in pre-computed tables.

Given an optimal quantization for a given localization scheme, we will need to inspect how the localization error behaves compared to the non-quantized case. A further impact of quantization is that by performing it, we are restricted on how many distinct points in the 2-dimensional space continuum we can localize to. Hence, the localization has to be assessed both in terms of localization error as well as in terms of localization coverage over a given area.

1.5 Thesis Structure and Contributions

The remainder of this thesis is organized as follows: Chapter 2 introduces related work on the topic of RSSI-based localization and k-NN localization in particular by providing three members of the k–NN family of schemes. We provide a brief introduction to quantization as the method to build the storage-based approach for localization using lookup tables. Search strategies for the relevant optimization problems are also reviewed here and used

in subsequent chapters. In Chapter 3 we introduce the datasets used in this thesis and what is the typical performance of the k-NN localization schemes for the datasets considered. We thus establish a baseline localization performance when no quantization is used and determine a k value for k–NN to use in the subsequent chapters. In Chapter 4 we present the 1-bit quantization localization and coverage performance are presented. We explain the methodology used to determine the best quantization based on a fingerprint database and an ensemble of points to localize and how we subsequently use potentially different sets of unknown points to test the quantization performance. We find the benefits to localization error when using local, i.e., per-peg quantization thresholds, and the poor performance when the quantization is attempted on RSSI values that were acquired in a modified environment. In Chapter 5 we move into multiple-bit quantization. The search strategy for the optimal quantization switches from genetic algorithms to Tabu search to deal with the increased search space. The multi-bit localization results are presented, primarily using 2-bit results. A comparison of 2-bit vs one-bit localization coverage reveals that the different members in the k–NN class of localization schemes exhibit different coverage improvement as more bits are added. Also, while the localization error is reduced with more bits used, it reaches a point of diminishing returns. Chapter 6 concludes the thesis summarizing the results in comprehensive plots and provides a review of shortcomings and threats to the generality of the reported results. Future extensions and research directions, including 3-dimensional cases and non-k–NN localization schemes are outlined. Some early results of the research described in this thesis were published in the 2019 IEEE International Conference on Communications (ICC 2019) [29].

In summary, the contributions of this thesis are:

- a process, facilitated by appropriate search techniques, to determine dataset-specific quantization of RSSI data for fingerpint-based localization schemes based on a fitness maximization objective,

- a study of the behavior of exemplars from the class of k-NN based localization algorithms, treated as a "black box" for the purposes of the fitness maximization objective,
- a study of the comparative advantages of different degrees of quantization across all considered k–NN schemes, with different datasets used for the optimization step and different ones for testing,
- the introduction of a coverage metric to describe the relative ability of combinations of quantization strategies, along with localization algorithms, to cover the area on which localization is to be performed,
- a comparison of the localization performance of the presented dataset-specific quantization against fixed quantization strategies.

Data and related code are available at: `www.github.com/uofanetdev/quantRSSI`

Chapter 2

Related Work

In this chapter we review the most relevant topics to the current thesis. First is the topic of localization and, specifically, data-driven approaches to localization, i.e., using schemes such as those based on k–Nearest Neighbor, that do not involve an intermediate step of producing a model/function to translate signal measurements to distances. Second is the topic of quantization, introduced in its general form, and then applied to the case of localization. We also describe the benefits, first introduced in Chapter 1, of having a quantized representation of received signal strength values. Finally, we briefly introduce techniques related to the search over large search spaces for optimization problems for which no efficient algorithms are known. Because those search schemes are "textbook" material, and used as a tool in the thesis, we provide only a limited discussion about them.

2.1 Indoor Localization

Although GPS does not work well in indoor environments, there are still compelling use cases for accurate indoor localization. One canonical example is a large storeroom where each item is tagged with a wireless sensor tag. As items may be moved around, and effort is expended in locating them, automated localization is beneficial. Another example is assisting the navigation of individuals, ostensibly equipped with smartphones where each smartphone can be assumed to be a tag. In the latter example, navigation is inside an, un-

known to the user, building and hence accuracy is an important attribute. In the first example, what is localized are the tags affixed to each item in the storeroom. Consequently, one is concerned both with issues of cost, as the storeroom example implies that each item has attached to it its own device. Equally, cost concerns may also limit the particular dedicated circuitry even in the case of smartphones. As such, the Received Signal Indicator (RSSI) is the most widely available measurement since it involves no cost because it is already possible to measure using the smartphone's WiFi (and other radio frequency technologies present on the smartphone). Note that, as with the case of the storeroom where many thousands of items need to be localized, a large indoor space is also possible to involve thousands of users and their smartphones. We are interested in localizing, efficiently and at little to no additional cost, large populations of items/users. A general comment applicable to the reviewed literature about localization techniques is that it, quite naturally, defines the localization from the point of view of a single to-be-localized tag/device. That is, to our knowledge, there is no literature that addresses the question of *how to scale localization to concurrently localizing large number of devices*. We therefore adopt a critical view of localization through the lens of not only limited cost vis-a-vis hardware requirements but also the ability to perform it *at scale*.

2.1.1 Localization Techniques

Localization techniques take a variety of forms. A recent survey [3] of localization techniques groups existing schemes to range-based and range-free as well as anchor-based and anchor-free. Anchors refers to the existence of nodeswhose coordinates are known a-priori. Note that anchor-free localization techniques provide relative node locations that, generally, do not possess a unique mapping to the physical space coordinates.

Another separation is between range-based and range-free localization. Range-based methods attempt to map signal strength values to specific dis-

tances, via a function – usually based on a path loss model of wireless transmissions. Subsequently, they apply a variety of multilateration approaches to provide a location for the tag that satisfies the distances derived from the first step. Due to multipath fading and shadowing inherent to the indoor environment, the parameters used in range-based models are site specific, i.e., not general, and are also often fragile, i.e., they assume that the model is insensitive to noisy measurements although there is evidence that the measurements used are often too noisy. Angle-of-arrival and time-of-arrival based approaches [17] can be lumped in the range-based schemes, but additionally require special hardware, usually in the form of antenna arrays and/or extra timing synchronization capabilities.

Range-free methods do not attempt to map signal strength to distance via any specific function, since it is well known that signal strength is a very noisy measurement and not necessarily monotonic with respect to distance due to the propagation environment complexities. There is also a growing use of the term model-free vs. model-based localization where model-free means that no explicit attempt is made to construct an underlying model for how the signal strength vary or relate across space. Model-based are the opposite and they include the range-based techniques, but may be also based on more advanced machine learning models.

The focus of this thesis are model-free, range-free, anchor-based schemes. The latter (anchor-based) is necessary because we are interested in mapping the locations of tags to absolute coordinates, rather than relative ones. This is a reasonable assumption since we consider the existence of RF emitters, like WiFi access points (APs), placed in advance at known locations. Because the transceiver hardware used for an anchor can be, technically, identical to that of the device we wish to localize, we use the term *pegs* for the anchor nodes and the term *tags* for the devices we wish to localize.

Profiling Techniques

A set of localization techniques that satisfy the model-free, range-free, anchor-based requirements are, the so-called, profiling techniques, also known as fingerprinting techniques [11]. Profiling techniques employ an a-priori data gathering phase, to associate with (known) locations in space, signal strength measurements from a number of known pegs, e.g., WiFi APs. Profiling schemes make no attempt to build an underlying model of how signal strength varies across space.

For instance, we can attempt to map the wireless indoor environment by collecting wireless fingerprints across the environment. Generally, these fingerprints are then stored in a database that maps a wireless fingerprint to a specific location. Devices can then be localized by comparing their received wireless signals to those in the database to estimate a likely location [11]. Perhaps the simplest technique in this category is based on proximity detection. Here, pegs/beacons are deployed in the environment in a dense pattern. Then, the location of any given tag is estimated to be the same as the location of the peg whose transmissions are received with the strongest signal (assuming the tag receives transmissions from more than one peg) [5]. Generally, the pegs used in this approach rely on a short-range transmission technology, such as infrared (IR) or radio-frequency identification (RFID).

Fingerprint collection requires significant effort and this is often seen as a strong disadvantage compared to, e.g., range-based techniques. As of now, most proposals on fingerprinting techniques, silently assume that the collection of the fingerprints in a database requires manual effort to map out the entire space of interest. Furthermore, the quality of the fingerprint database will affect the quality of localization. For example, the resolution of the database (i.e., the distance between two adjacent fingerprints in the database) will directly affect how accurate the localization can be. Figure 2.1 illustrates the process of fingerprint localization. Fingerprints are collected and placed in a database offline. In the online phase, a wireless tag sends its received

RSSI values to a central server which performs localization. The estimated location is optionally returned to the wireless device.

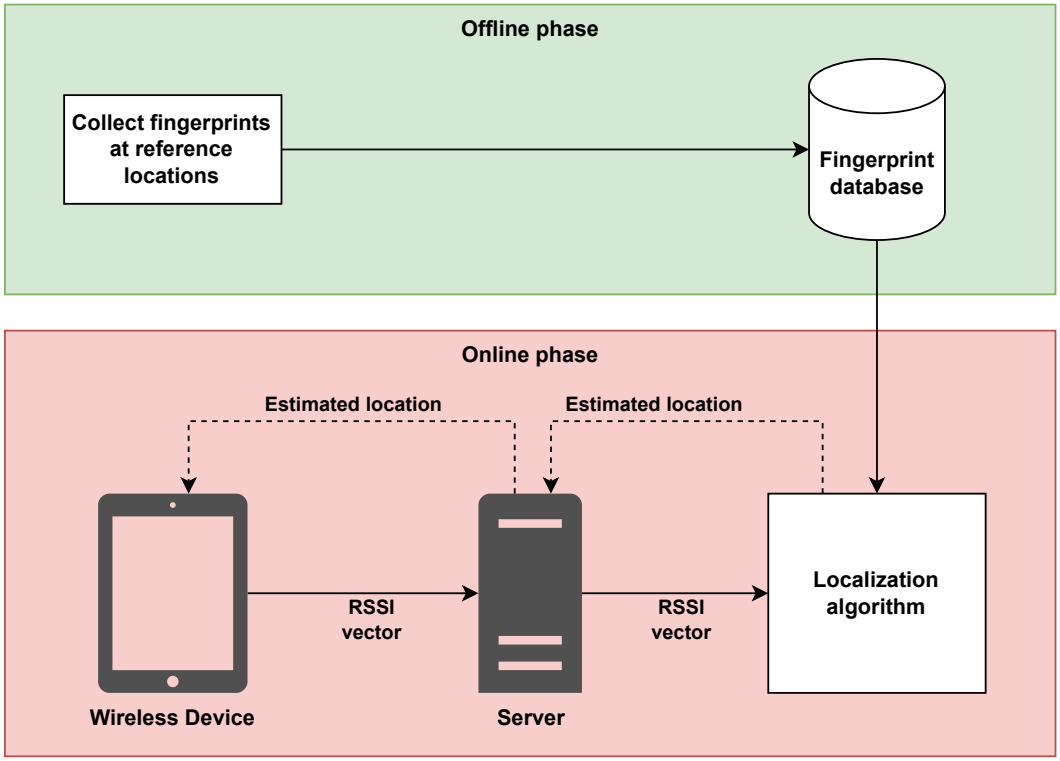


Figure 2.1: A generic fingerprint-based localization.

2.1.2 Received Signal Strength (RSS)

One easily obtainable metric influenced by the distance between transmitter and receivers is the (wireless) received signal strength. There is an abundance of devices fixed at specific and known locations, namely WiFi APs, whose transmissions, received by a receiver at some point in space will depend on the distance to the APs. RSS measurements do not require any specialized hardware to produce. The signal strength versus distance, d, relationship is often described through path-loss models relating the received power to be proportional to $1/d^a$ where a is the *path loss exponent*. There exists no single universal path loss exponent for all indoor environments, and it is influenced by the exact pair of transmitter/receiver and the nature of the obstacles in the environment, e.g., see the work carried out in [15]. It is for this reason that the

derivation of a model from RSS measurements is considered as error-prone. This leaves fingerprinting techniques as a viable alternative. Note also that RSS measurements are referred to as RSS Indications (RSSI) by the various device manufacturers and we will conflate the two (RSS and RSSI) in the rest of this thesis as they mean, from the operational point of view, the same thing.

RSS-fingerprinting techniques rely on collecting a database of RSS values at known locations. That is, a set of (known) locations are mapped to their corresponding RSS vectors. We will often use the term *signal space* to refer to the RSS measurements at a location in physical space. Conceptually, it is assumed that there exists a one–to–one correspondence between the two types of spaces. Mapping from the signal space to physical space is the task of the localization technique. From now on, we use the term "space" for the physical space and qualify it as "signal space" when referring to the RSSI values corresponding to a (possibly unknown) position in physical space. Note also that the one-to-one correspondence between the two spaces is a desirable property, but not necessarily attainable using any specific localization scheme.

While we do not endorse the generation of a model out of the RSS fingerprints, one can think of the profiling stage, i.e., the fingerprint collection, as a collection of a "training" set. When localization of a tag at an unknown location is attempted, it does so by presenting the vector of RSSI values it sees at that point and we are asked to determine the location in the physical space. This, latter, part can be thought of as a "testing" phase of the localization.

The degree of noise present in RSSI values and the perpetually changing RF propagation environment discourage any hope that the localization scheme is able to tease out the noise as well as to model the RF propagation adequately to account accurately for the impact of the physical environment. Even if this were to be the case at one instant in time, changes to the RF propagation environment might mean that the same physical location can correspond to a different signal space measurement at the next instant. Thus any deterministic localization scheme, to produce a new localization result for

the same signal space vector would need to be explicitly informed that the RF environment has changed (and how it has changed), and have its database of fingerprints updated.

A final comment about profiling schemes in general, but outside the scope of the current thesis, is the "normalization" of RSSI measurements carried out, by a heterogeneous set of devices. There exist proposals, such as one based on the Procrustes method outlined in [31] to map measurements of heterogeneous devices to a common "standardized" form. In this thesis, the datasets we consider involve homogeneous devices, and therefore we do not consider an intermediate standardization step. However, in a real deployment, the use of measurements in a common fingerprint database coming from a heterogeneous set of devices can lead to poor results if not properly pre-processed to a common standardized/normalized form.

2.1.3 k-Nearest Neighbours (k-NN)

One of the earliest group of techniques using fingerprint databases involves k-nearest neighbours (k-NN). The common feature of those algorithms is that an RSSI vector from a tag at an unknown location is compared to entries in the fingerprint database and the k closest entries in signal space are returned. The distance between signal space vectors is defined to be their Euclidean distance. That is, assuming there are n pegs, and a tag measuring an RSSI vector $r^{(u)}$ at an unknown location, its distance to a fingerprint RSSI vector, r, is defined as:

$$d = \sqrt{\sum_{j=1}^{n} (r_j^{(u)} - r_j)} \tag{2.1}$$

Notice that: (a) it is tacitly assumed that the vectors have (meaningful) values for each of their elements, while in a real setting some pegs' signals may not be received at all, and hence there could be missing values (an issue we will address in subsequent chapters), and (b) the precise units, like e.g., dBm, for

the RSSI values are irrelevant, since a distance can be defined no matter what units we chose. The k fingerprint entries for k–NN based schemes correspond to k known locations, and their coordinates in physical space are used to estimate the unknown location. In this thesis we consider three representatives of the k-NN family of localization schemes. The three schemes differ in the way they combine the coordinates of the k-NN.

2.1.4 RADAR

The RADAR algorithm was one of the first techniques used as an improvement over limited-ranged simple proximity sensors using IR [2]. Experimentally, RADAR involved setting up 802.11 APs throughout a large space. As is typical of a range-free localization technique, localization involves two phases. In the offline phase, a host moves about the space of interest, continuously sending UDP packets. The base stations then record the signal strength and timestamp of received packets. This data is then merged in a database, associating the 2-D physical location of the host (in terms of x and y) with the mean of the signal strengths from each base station for that location. Concretely, the database table contains entries with tuples <(x, y), r> where r is the vector with n elements (as many as the APs/pegs). In the online phase, devices are localized. Given a tuple containing signal strength from an unknown location, the k nearest neighbours are found using the Euclidean distance in signal space introduced previously. If we index the k neighbors with smallest distances with i, and denote with d_i the signal space distance between the unknown location's $r^{(u)}$, and the location of the i–th neighbor, then the location estimation produced by RADAR is:

$$(x, y) = \frac{1}{k}(\sum_{i=1}^{k}(x_i, y_i)) \tag{2.2}$$

Notice that RADAR does not take the magnitude of d_i into consideration. In the experimental study [2] it was reported that RADAR achieved a median

localization error between two and three meters, using three base stations spread throughout the floor of a building.

2.1.5 LANDMARC

LANDMARC was developed in an effort to localize using RFID technology [21]. Despite using a different wireless technology, the experimental setup closely parallels that of RADAR. RFID readers are installed throughout a space, acting as the "base stations" that receive RF signals from RFID tags. However, instead of having a discrete offline data-collecting phase with a mobile host in RADAR, so-called "reference tags" are placed at fixed locations throughout the space. These reference tags can thus act as neighbours in the online phase. The advantage of this approach is that the system is more robust to changes in the environment, as any changes that affect the localization of unknown devices should similarly affect the signal strengths received from the reference tags. As opposed to the simple average of location that RADAR used, LANDMARC uses a weighted average of k neighbours based on the signal space d_i distances. The smaller the distance the higher the weight. Additionally, the weights are all normalized, to ensure that the weights sum up to 1. In summary the location estimate in LANDMARC is as follows:

$$(x, y) = \sum_{i=1}^{k} w_i \cdot (x_i, y_i) \tag{2.3}$$

where each w_j is given by:

$$w_j = \frac{1/d_j^2}{\sum_{i=1}^{k} 1/d_i^2} \tag{2.4}$$

and where d_i stands for the Euclidean distance in *signal space* between the measurement to localize and the i–th of the k neighbors. It was reported in [21] that with 4 RF readers in a roughly 40 m^2 environment, the average localization error was one meter.

2.1.6 LEMON

Another weighted-average k-NN technique, LEMON, has been shown to work well with low-cost, low-power wireless sensors [9, 10]. The offline phase is similar to RADAR. The study in which LEMON was proposed deployed 915 MHz ISM band wireless sensors instead of WiFi APs. In the online phase, the location is estimated using a weighted average, with w_j calculated by:

$$w_j = \frac{d_{\max} - d_j}{k * d_{\max} - \sum_{i=1}^{k} d_i} \tag{2.5}$$

Here, $d_{\max}$ is the largest signal space distance across the k neighbors. The weight of the furthest neighbour is therefore 0. Effectively, only $(k-1)$ neighbours are used in the weighted average calculation of the location. The furthest neighbour is used to calibrate and normalize the weights of the remaining, neighbors. In various studies LEMON has been shown to have a small average localization error [9,10] although one could argue that to achieve such result, the pegs need to be densely deployed across the environment.

2.1.7 The Complexity of k-NN Techniques

While, trivially, the search for the k-NN neighbours of a given vector can be carried out sequentially, the option is unappealing for large database of fingerprints, and more efficient algorithms are available. A well-known algorithm for k-NN in *expected* logarithmic time is due to Friedman, Bentley, and Finkel [6] making use of k-d trees. However, its worst case execution is still linear, corresponding to "pathological" cases of data point distributions. It has been argued that pathological cases are rare in practical data sets. Similar is also the expected time complexity of the "vantage point" tree (vp-tree) algorithm by Yianilos [28] which is claimed to be suitable when the distribution of the data is from a lower dimensional space but embedded in a high-dimensional space. The lower dimension (physical) space mapping to high-dimensional (signal) space fits our localization use case. Other algorithms

have also been proposed for *approximate* results to k-NN queries but we do not consider them, as any approximate answer may have an additional negative impact on the, already approximate, localization results. In any event, our interest is in selecting a representative time complexity for k–NN queries, and this will be assumed to be $O(log(N))$ in keeping with existing literature. We consider as unimportant the computational cost for pre-processing (building), or the space cost for storing the database, as those are one-time (or rarely recurring) costs and their corresponding storage is plentiful.

2.1.8 Localization Coverage of k-NN Techniques

A common characteristic of all k-NN localization techniques is that, they are weighted summation of known coordinate pairs of the k profiling points selected. While not studying k-NN schemes specifically, Zanca et al. [30] noted that the best scheme of the ones they considered *"weighs the RSSI from the anchor nodes according with the reliability of such a reading, which decreases with the received power"* which is a statement on the need to consider the signal space distance when producing the weights for the determination of the location given known emitter locations (anchor nodes in their cases). We therefore expect that the weights used by LANDMARC and LEMON will produce better results compared to RADAR. This observation is confirmed later in this thesis.

Additionally, the k-NN based schemes produce localization results enclosed within the convex hull of the profiling point (x_i, y_i) coordinates. Interestingly, in all studies we encountered the profiling points are also "inside" the convex hull defined by the pegs. Some techniques include, in their profiling set, locations at are adjacent to, if not exactly at, the locations of the pegs. As a result, the possible localizations using weighted k-NN are within the convex hull of the pegs. By the same token, locations outside the convex hull of the pegs (or of the profiling points, as the case may be) *cannot* be localized using such schemes.

In the relevant literature an, often unstated, assumption is that the evaluation of the techniques is limited to localizing points within this convex "localizable" area, leaving outside the discussion, how can a device determine whether it is inside or outside the area. A conceivable answer to this technical requirement is that larger areas are split into "regions" and some form of gross localization takes place (even possibly as direct user input) to select which area the device is currently inside. Subsequently, area-specific localization can be performed. We will also follow the same assumption, by not considering as candidates for localization any points outside the convex hull. Additionally, the placement of APs, is such that APs are often placed at the corners of a large space, such that the resulting convex hull is a rectangular area.

Except for unlikely and unnatural cases of AP deployment, e.g. all being co-linear, any point of the convex hull can be expressed as the weighted sum of AP coordinates by solving to determine the specific weights. Conversely, the weights computed by any particular localization algorithm may not have sufficient variety to be able to produce the coordinates of any arbitrary point within the convex hull. It is therefore of interest, and addressed in the thesis, to also inspect how well a k-NN localization algorithm, together with a given fingerprint database, covers the convex area.

2.2 RSS Quantization

As RSS data are noisy due to myriad physical aspects, Gao et al. [7] demonstrated an attempt to reduce the amount of data transferred to convey the RSS information, while preserving a high localization accuracy. The benefit of their approach is the reduction of the data needed to be conveyed to an "edge" node responsible for the localization of tags. Thus, the tag measures the RSS from various pegs/APs, reduces the size to represent this information to a few bits, transmit those to the edge node, and the edge node executes the localization computation which, ostensibly, would have been too onerous for

the computational resources of the tag. Whether the tag is to be informed of its (estimated) location or whether this information is of value to some other application running at the backbone of the network, well-connected to the edge node, is irrelevant to the process of reducing the RSS values from all the pegs to a few bits. Note that the above view of the process involves at least a unidirectional communication between tag and edge node, and possibly bidirectional if the tag needs to know the localization result as well.

The reduction of the volume of data suggested by Gao et al. [7] employs quantization. In quantization, the RSS range is divided into n intervals, using $n-1$ boundaries. The corresponding intervals have mid-points, that are the representative RSS values for the interval. For each *measured* RSS value, the tag identifies in which interval it corresponds and replaces it with the representative RSS of the interval. However, since the quantization is known to the receiver (e.g., the edge node) as well, what is needed to convey the representative RSS is just an index value indicating which interval the measured RSS falls on.

What was argued in [7] is that since the distribution of RSS measurements is non-uniform, higher granularity might be desired in certain regions of the RSS range, and less granularity in other regions. If, for example, the boundaries are concentrated in one narrow range, then, once quantized, the RSSI values in that range will be more finely represented. Note that the quantization step is agnostic to what localization algorithm is used, so the localization scheme can be treated as a black box, even though Gao et al. [7] examined the performance of the quantized representation just for LEMON [9]. The notable result is that they find four intervals (2 bits to index them) is sufficient.

2.2.1 Pre-computed Quantization Intervals

Some more recent work into RSS quantization has also been performed on pre-computed quantized levels. That is, instead of using an elaborate process to determine the best quantization, the quantizer formulas used are fixed

and indifferent to the distribution of measured RSS. Specifically, Khandker et al. [16], evaluate four different quantizer formulas which operate over an arbitrary number of bits. Assume E_{max} and E_{min} are the maximum and minimum of the range of values to be quantized. The following pre-computed quantization schemes are defined:

$$\text{(linear quantizer)} \quad f_1 = (E_{max} - E_{min})/(2^n - 1) \tag{2.6}$$

$$\text{(square root quantizer)} \quad f_{2(i)} = (E_{max} - E_{min}) \times \frac{\sum_{j=1}^{i} \sqrt{2^{j-1}}}{\sum_{j=1}^{2n} \sqrt{2^{j-1}}} \tag{2.7}$$

$$\text{(cubic root quantizer)} \quad f_{3(i)} = (E_{max} - E_{min}) \times \frac{\sum_{j=1}^{i} \sqrt[3]{2^{j-1}}}{\sum_{j=1}^{2n} \sqrt[3]{2^{j-1}}} \tag{2.8}$$

$$\text{(logarithmic quantizer)} \quad f_{4(i)} = (E_{max} - E_{min}) \times \frac{\sum_{j=1}^{i} \log 2^{j-1}}{\sum_{j=1}^{2n} \log 2^{j-1}} \tag{2.9}$$

f_1 describes a linear quantizer. In this case, the quantizer divides the signal space between E_{max} and E_{min} into 2^n equal areas. The other 3 formulas divide the signal space more finely towards E_{max} with fewer divisions towards E_{min}. They found that 4 bits was enough to achieve a localization accuracy approaching that of full-resolution data, with f_1 and f_4, the two formulas closest to a linear quantizer, providing slightly better localization performance than the other two.

Mizmizi and Reggiani [19] attempt to quantize RSSI values that are resulting from simulations using a path loss formula and they consider a contrived 2-dimensional grid. They do not attempt to determine an optimal quantization, but rather the relation between the tags ("beacons" in their nomenclature) and the number of quantization levels to achieve a given mean square

error. What is notable is that they seem to reach the best number of beacons for a given MSE at a quantization of between 6 and 10 levels (roughly between 2.5 and 3.2 bits). Earlier, Patwari and Hero [22] were probably the first to link the quantization and the proximity (really, single-bit quantization) to fundamental limitations due to the Cramer-Rao bound, and provided evidence that 3 bits (8 levels) may be sufficient, but their results are based on simulations employing a path loss model.

2.2.2 Data-driven Quantization

The quantization options described in the previous subsection do not adapt to the actual fingerprints collected. Yet, the work in [7] suggests that the quantization should be driven by the actual fingerprint set. We are therefore posing the question as follows: for a given set of fingerprint data, what is the *optimal quantization* of those values (and hence of any unknown measurements we will subsequently wish to localize)?

The thrust of the research described in this thesis is the determination of the optimal boundaries of a set of RSSI values (the "training set") which produces the optimal values for the boundaries. We consider b-bit quantizations, which is equivalent to determining $2^b - 1$ boundaries. Given that the original domain of the RSSI measurements is also limited in terms of the bits, say m-bits large, the determining a best boundary for a 1-bit quantization of those values is computationally simple, since the boundary we are after is one of the 2^m values. However, the search for the boundaries becomes exponentially more difficult as more boundaries have to be determined. Even for a single boundary value, we need to capture whether a chosen value is better or worse option than another value via a metric which captures how it would result in the localization not for a single unknown point but for the localization of an ensemble of points. The search process has to be generic enough that the entire localization algorithm can be treated as a black box. We address this need in Section 2.4.

2.2.3 Global vs. Local Quantization

A crucial step in the quantization process is whether the boundaries that define the quantized levels should be **global**, i.e., the same values across all pegs or **local**, i.e., different boundary values for each peg. Given the vagaries of the wireless propagation environment, it might be beneficial to define per-peg, i.e., local, boundaries. The time invested in searching for the best boundaries would then be increased. For example, for a 1-bit quantization across n pegs, there are n local boundaries instead of only 1. This means that further attention needs to be paid to the optimization technique (really search techniques) employed during the determination of the optimal quantization.

2.3 Applications of RSSI Quantization

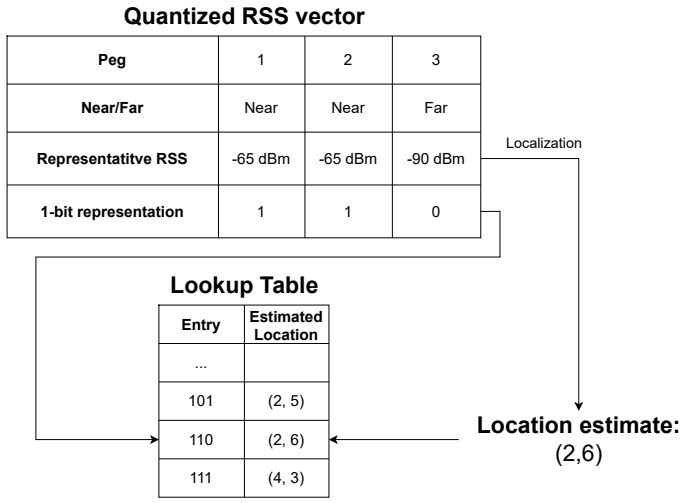


Figure 2.2: Populating location estimates in a lookup table using 1-bit localization.

One benefit of quantization is the ease in which estimated locations can be pre-computed and stored in a lookup table, for offline access. This can be illustrated using a 1-bit quantization example as shown in Figures 2.2 and 2.3. The example threshold/boundary is assumed to be global and equal to -80 dBm. The two bins (zero for lower values, one for higher values) have representative RSSI values equal to -90 and -65 dBm correspondingly. Figure

2.2 captures the offline lookup table creation. Since the signal from each peg is represented by a single bit[1], the concatenation, in a fixed order, of the n bit values (for n pegs) can represent, collectively, all the signals to n pegs. There are 2^n possible such bit vectors and each value can be thought of as an index to a lookup table. The table is populated with localization outputs using the given (e.g., a k-NN) localization algorithm. This can be done once using 2^n invocations to the localization algorithm. Since each of the intervals has a representative value, there are actual RSSI values that can be used by the localization algorithm to produce those location estimates.

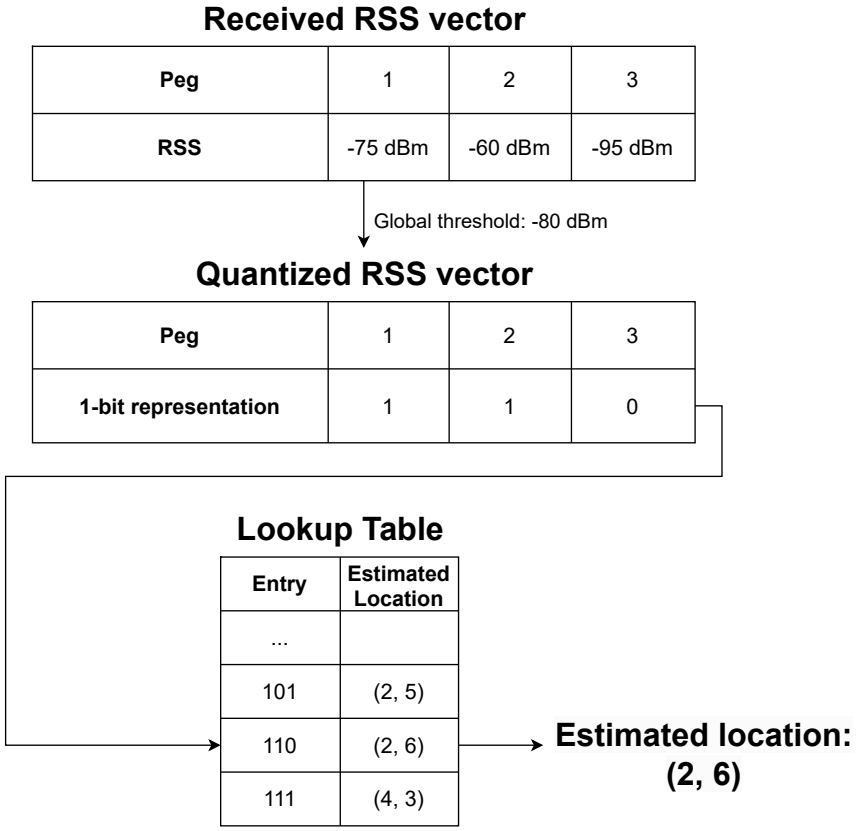


Figure 2.3: Retrieving 1-bit quantized location estimate from a lookup table.

The table produced as shown in Figure 2.2 is used in Figure 2.3. An unquantized ("full resolution") RSSI vector is measured by a tag device, whose location is unknown. Using the quantization threshold(s) it computed in ad-

[1]In the special case of 1-bit localization the value of a zero is a "far" reading (lower than quantization threshold) and the value of one is the "near" reading.

vance, it can convert each RSSI value into a 1-bit value. If a given RSSI value is below the threshold, then 0 is used, else 1. Given a fixed a-priori order of the pegs, and hence of the bits produced by this quantization, a bitvector is produced which acts as an index to the table calculated as in Figure 2.2, retrieving the estimated location of the tag. Note that the tag can just listen to the peg transmissions, and assuming it is equipped with the table and with knowledge of the threshold(s) to perform quantization, it does not need to transmit anything to determine its location.

With a larger number of bits, b, used for quantization a total of 2^{bn} locations will need to be stored in a pre-computed lookup table. Ideally, this number will be small enough that a tag device can store it in its own onboard memory. This way, the tag device can estimate its own location offline, in constant time, without the need to store the fingerprint database. However, for large 2^{bn} the necessary storage for the lookup table can become comparable to the contents of the entire fingerprint database.

2.4 Optimization Techniques

The challenge with quantization comes from the search for the boundaries that results in the optimum quantization. The optimality is judged by means of maximization of the fitness function. By way of example, if the RSSI values are 7-bit values, there are 128 distinct values for a single boundary. Thus, the search space for boundaries increases rapidly with the number of bits used for quantization. With 2-bit quantization, there are $\binom{128}{3} = 341376$ combinations of three boundaries. With 3-bit quantization, there are 94 million different combinations, and so on. Thus, an efficient search method is needed. In this thesis we have employed two search techniques whose operation is summarized here: a Genetic Algorithm (GA) for 1-bit quantization, and a Tabu search scheme for multiple-bit quantization.

2.4.1 Genetic Algorithms (GAs)

An elitist genetic algorithm (EGA) [24], as used in [7], can be used to determine an optimal threshold when quantization into two values (1-bit) is desired, i.e., to determine a threshold value to separate “near” and ”far” values. GAs are a set of evolution-inspired algorithms that produce solutions to optimization problems. Candidate solutions, which in our case are quantization boundaries, are encoded as chromosomes. Then, for each generation the following happens:

1. *Selection*. Choose two parents. These can be chosen from roulette selection, where more “fit” chromosomes, as per the fitness function, have a higher probability of being chosen, through there are other selection methods such as tournament selection.

2. *Crossover*. With some probability, a crossover event may occur. In the simplest case, a single point of crossover is used. That is, some part of the chromosome is chosen to be the crossover point. Up to that point, for one child chromosome, the “genes”, or boundary values are chosen from parent 1. After that point, the genes from parent 2 are used. The complementary child chromosome (parent 2 then parent 1) is also created. If no crossover point is used, then both parents are used as-is.

3. *Mutation*. With some probability, each gene has a chance of mutating to some random value. This mechanism forces the algorithm to try new solutions, i.e., new boundary values in our case.

The choice of fitness function is important. Intuitively, at each generation, chromosomes which are more “fit”, that is, a better solution to the problem at hand – are more likely to carry forward to the next generation. We use the fitness function described by Equation 3.1. An EGA preserves the most fit individual across each generation. This prevents the most fit individual from being altered by crossover and mutation. Note that to derive the fitness function value, we have to try to process (localize) an ensemble of points to derive

their, collective, mean and standard deviation of localization error. That is, each candidate quantization, requires that we run the black box localization for each point from a given set of testing points. Thus, we will be distinguishing between a testing set and a "training" set – the latter being the set of fingerprints and their associated, known, locations.

The initial population of chromosomes is randomly generated. Namely, for each boundary in each chromosome (of a set of chromosomes, which is a parameter), a random value from the set of all possible boundaries is chosen. The set of possible boundaries is informed by the RSSI values contained in the fingerprint dataset. First, all unique RSSI values from the dataset are ordered in terms of increasing value. Then, a possible threshold value is calculated using the midpoint of any two successive RSSI values present in the fingerprint database. Thus, if there are N unique RSSI values in the fingerprint dataset, there will be $N-1$ possible threshold values.

2.4.2 Tabu Search

EGA is, computationally, convenient for local 1-bit local quantization but its performance deteriorates particularly when performing local multi-bit quantization as it has the tendency to become trapped in local minima. Compared to 1-bit quantization, there appear to exist too many local minima in multi-bit quantization, from which GA can only escape very slowly. Thus, we also employ Tabu (or Taboo) search [8], a local search method that forces the exploration of new solutions by performing occasional "jumps" across the search space.

At its core, Tabu search is an elitist greedy local search algorithm. However, it escapes the trap of local minima by using the, so called, Tabu list, of candidate solutions. In simple local search algorithms, such as simple hill climbing, a list of "neighbours" is generated at each iteration[2]. A neighbour to

[2]Not to be confused with neighbours in signal space or physical space, this is a third possible meaning for neighbour. Here, neighbour means a numerically similar candidate solution, i.e., one with slightly different values for some (possibly even one) boundary.

a particular (current) solution is created by altering a single boundary value for any particular peg, resulting in a new candidate solution that is different from the current solution in only one boundary value. A high-level pseudocode for the generation of the neighbour candidates is presented in Algorithm 2. MAX_NEIGHBOURS is a parameter to limit the size of the number of neighbours considered. Consistently with the GA case, the fitness function provides a direction for which solution to explore next. Specifically, the neighbour with the highest score is explored. However, this approach has the tendency to become stuck in local maxima, or in regions where many neighbours are equally suitable. Tabu search avoids this situation by forcing the algorithm to deliberately check "worse" neighbours.

The high-level pseudocode for the Tabu search as we used in Chapter 5 is shown in Algorithm 1. It utilizes a so-called short-term memory structure, the eponymous Tabu list. This structure holds a list of recently checked neighbours, and prohibits the algorithm from analyzing them again. This list holds each item only temporarily, so at a later time during the execution of the search, these neighbours may be checked again, hence the short-term nature of the memory structure. By exploring the neighbours of these weaker solutions, the hope is that globally better solutions can be found. Generally, this Tabu list is implemented using a fixed size, and old entries are evicted from the list when the structure is full. In addition, longer-term memory structures can be used, such as more lists with different expiration schemes. The goal for short-term memory is to encourage exploration of the immediate local area, while longer-term memory structures are used to force exploration of more distant areas in the search space. Traditionally, the Tabu search approach has found success in combinatorial optimization problems like scheduling, as well as more classical problems such as the travelling salesman problem and graph colouring.

Note that the size of the Tabu memory structure will necessarily increase if local quantization is used instead of global quantization, and it if more than

Algorithm 1 Tabu search algorithm using Tabu list for short term memory and random restarts for long term exploration

$n_{best} \leftarrow$ Initialize best neighbour with random values
last_fitness $\leftarrow$ fitness(n_{best})
last_change $\leftarrow$ 0
while runs - best_found < min_runs **do**
  while i < maxIter **do**
    Increment i
    if i - last_change > long_term_threshold **then**
      $n_{best} \leftarrow$ Initialize best neighbour with random values
      last_change $\leftarrow$ i
    end if
    $\mathcal{N} \leftarrow$ generate_neighbours(n_{best})
    bestNeighbour $\leftarrow n_0$
    for $n \in \mathcal{N}$ **do**
      Quantize (local or global) the testing set according to n
      Localize quantized training set
      Evaluate fitness(n) for the quantized training set
      if fitness(n) > last_fitness **then**
        last_fitness $\leftarrow$ fitness(n)
        last_change $\leftarrow$ i
        add n to tabu list
        if fitness(n) > fitness(n_{best}) **then**
          $n_{best} \leftarrow n$
          best_found $\leftarrow$ i
        end if
      end if
    end for
  end while
end while

Algorithm 2 generate_neighbours(n_{best})

```
 1: i ← 1
 2: bound_order ← shuffle(0, 1, ..., num_bounds)
 3: neighbours ← empty list
 4: neighbours[0] ← n_best
 5: while i < MAX_NEIGHBOURS do
 6:   for b ∈ bound_order do
 7:     neighbours[numNeighbours] ← n_best
 8:     neighbours[numNeighbours][b] ← random_bound
 9:     i ← i + 1
10:     if i > MAX_NEIGHBOURS then
11:       return
12:     end if
13:   end for
14: end while
```

one, say b, bits are used by the quantized representation. Each of the n pegs would require the determination of $2^b - 1$ boundaries, hence a candidate solution involves $n(2^b - 1)$ variables. The canonical approach for short-term memory is to use a Tabu list, and we implement this list using a ring buffer, where older values are continually evicted as new neighbours are entered. For long-term memory, however, there can be a variety of approaches used. As stated previously, the goal for long-term memory is to force the algorithm to explore distant areas. In our case, we chose to implement the long-term memory as a simple reset rule, similar to what is done with local search techniques that utilize random restarts. This approach appeared to work well in practice.

Chapter 3

Datasets & Baseline Results

In this chapter, we describe the used datasets and provide baseline localization performance results under no quantization. The two datasets we examine are: one in a carpark, and one in a library.

3.1 Carpark Dataset

The carpark dataset, was previously used in [7] to examine how few bits are needed to achieve a similar level of localization accuracy as with full resolution for the RSSI measurements. In that study, they found that two bits was sufficient.

The dataset was collected in a 12×8 m^2 space in the lower floor (underground) carpark, a relatively quiet RF environment. Using CC1100 devices [25] in the 915 ISM MHz band, 10 peg devices were laid out surrounding the rectangular area in a grid (see Figure 3.1. Within the grid, RSSI measurements were collected roughly every 1 meter apart in both directions. The side of each grid cell is 2 meters, pegs are represented by black squares. Measurements in the fingerprint database are taken from the locations with solid black stars. The to-be-localized measurements are collected by a tag placed at the locations indicated by circles. When a car is present, to play the role of the obstacle, we indicate, in the same figure, the three alternative placements that were used. Note that Figure 3.1 exaggerates the size of the placed car and does not properly reflect the scale of the car relative to the space. Thus, for the

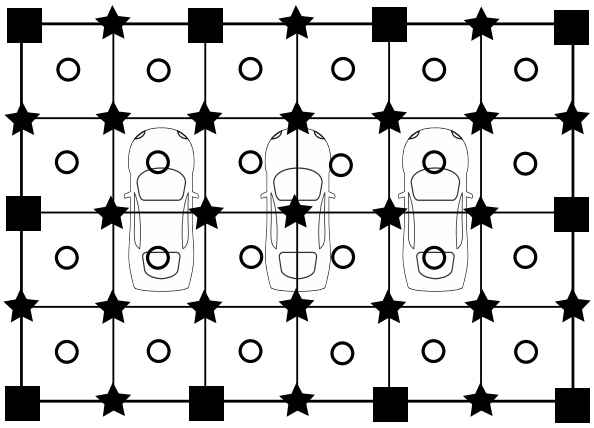


Figure 3.1: Layout of the carpark environment (not to scale).

same placement of pegs, there are three subsets of measurements: one with no car, and three with a car present in the left, center, or right of the space. For each of the subsets, we have a set of fingerprint (star) measurements and a set of tag to-be-localized measurements (circles). In all cases, despite the presence of the car, it was possible to have 25 fingerprint measurements and 24 tag to-be-localized measurements, with the exception of when the car was placed to the right where the tag measurements were 23 instead of 24 due to the obstruction caused by the car.

3.1.1 Post-Processing

With the exception of a few[1] of RSSI vectors that had missing values (peg too far for a measurement to be possible) the carpark dataset had complete RSSI vectors across the fingerprint and the tag datasets. This is a side-effect of the dense peg deployment. The vectors with missing values were eliminated from the dataset because it was deemed that, due to being just a few, their removal would not impact the localization results in any notable way.

[1]Five RSSI vectors, all in "Car to the Left" (CL) layout, have missing peg measurements.

pegs received	RSSI vectors
1	908
2	415
3	84
4	12
5	1
6-13	0

Table 3.1: Simultaneous peg measurements in RSSI vectors (library dataset).

3.2 Library Dataset

The second data set was created using 13 iBeacon devices connected by the Bluetooth Low Energy (BLE) protocol, with data being collected using an iPhone 6S in the Waldo Library of Western Michigan University [18, 20]. The library areas considered measures 200 x 180 feet (60.96 x 54.86 m), with each "cell" of the map in Figure 3.2 corresponding to a square of size 10 x 10 feet (3.04 x 3.04 m). iBeacon devices, which are used as the "pegs," are deployed in the 13 numbered circles, and the signal strength of each iBeacon is collected using a mobile device by a person standing in the cell.

As in the car park dataset, the area is divided into a grid. However, the iBeacons are not at the edge of the area, but interspersed throughout the library. In addition, due to the low range of the BLE devices, at many areas in the library, not all iBeacons can be seen. As such, the dataset is relatively sparse. Using this dataset will allow us to examine the effect of quantization when the peg density is smaller, and the effect of many missing values from the RSSI vector collected by the tag/phone due to the distance from the pegs. Indeed, the tag was never able to receive signals from all 13 pegs at once. At most locations, only one of the tags could be received, with a maximum simultaneously received being just five. Table 3.1 shows this distribution.

Across the fingerprint data points, the RSSI from the received pegs ranges between -88 to -55 dBm. The histogram of their values is shown in Figure 3.3.

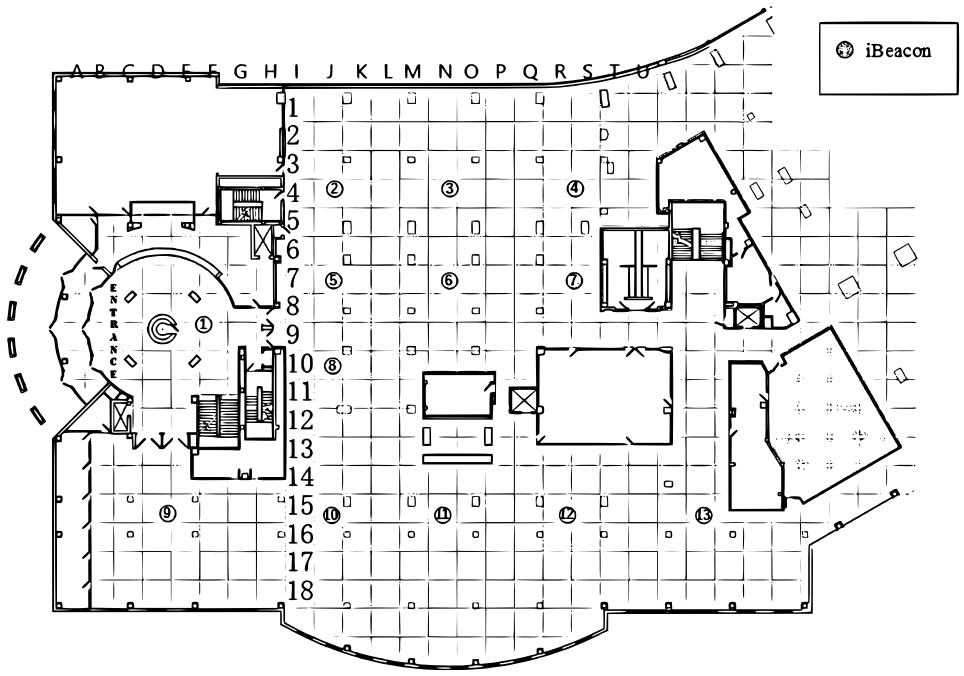


Figure 3.2: The library area, from [18, 20].

3.2.1 Post-Processing

Contrary to the carpark dataset, the library set is limited in two ways. First, there are no dynamics in the environment that we know of, i.e. information about obstacles being present and or being moved during the data collection. A more important concern was the need to produce imputed values for the missing measurements. Since all vectors had some peg measurement missing, it would not have been possible to just drop them from the dataset.

The imputed value should be a small RSSI value to indicate that the point is "distant" from a peg, and smaller than any received RSSI signals. At the same time it should not be much smaller than needed because it could, under a k–NN setting, severely influence the weights used in combining the k fingerprint locations. Across all points (fingerprint or tag locations) the weakest signal recorded was -111 dBm. With this in mind, we use -120 dBm as the imputed value. The value of -120 dBm is also very likely close to the noise floor.

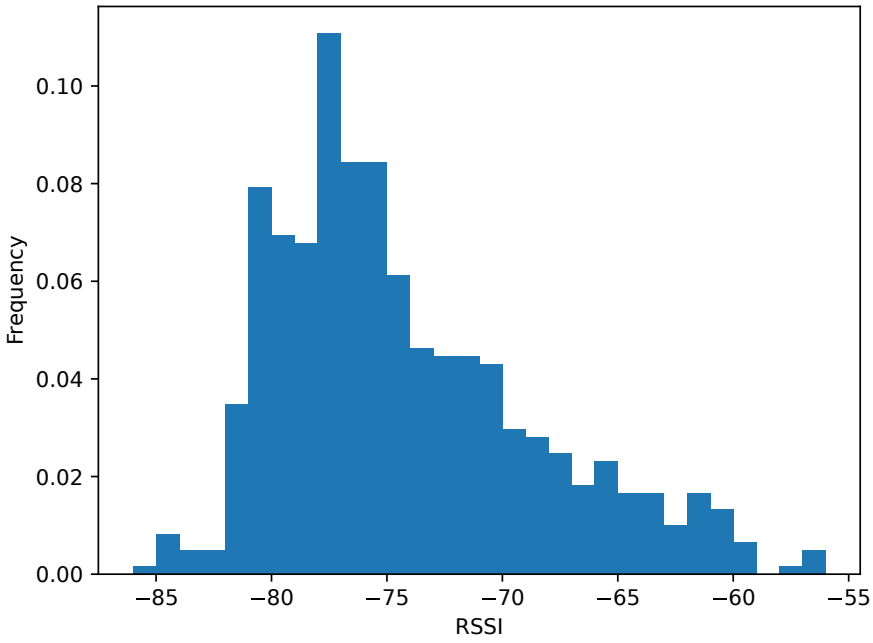


Figure 3.3: Histogram of fingerprint RSSIs across all pegs (library dataset).

3.3 Other Datasets

Additional datasets existed at the time this thesis was undertaken but were eliminated from consideration for the correspondingly stated reasons.

3.3.1 UJIIndoorLoc

The UJIIndoorLoc dataset [14] is comprised of over 21,000 samples covering 3 buildings of Universitat Jaime I with 4 or more floors each. The localization area is almost 110,000 m^2. Within this dataset, location is described using longitude and latitude, while floors are described using an integer for floor number. Each data point consists of a vector of RSSI values from 520 possible WiFi access points, collected using a variety of Android devices. The actual RSSI value varies from -104 dBm to 0 dBm. The dataset is further classified into room type (e.g., office, classroom), the user collecting the data, as well as the type of device. UJIIndoorLoc was eliminated as the dataset is just as "sparse" as that of the library. That is, at most of the measurement locations,

only a small number of signals from pegs can be received. Many of those points would have to, again be imputed RSSI values, again suggesting that it would not add anything more than what the library dataset alredy provides.

3.3.2 AmbiLoc

The AmbiLoc dataset [23] represents a lengthy period of observations of signal strengths based on various RF emissions. It is a collection of 23 fingerprinting "sessions" taken around 3 floors of an office building, 2 floors of a university building, and an apartment unit across a year of time. Instead of having fixed "peg" devices, they use ambient signals in the FM, GSM-900, or DVB-T bands and measure signal strength every 10 Hz within that band. While rich in terms of duration, it lacks information as to when changes were happening in the environment which could allow us to determine points in time when the propagation environment would, for a fact, have changed (which we can do in the carpark dataset). For those reasons, i.e., that it is multimodal and no information about the changes occurring during the measurement period are available, it was not used

3.4 k-NN Localization Tuning

With k-NN localization algorithms, as described in Section 2.1.3, the most salient (and only) parameter that needs to be chosen is k. The larger the value of k, the more neighbours that are chosen from the fingerprint database to estimate the location of any particular unknown location of a tag. If the value of k is too small, then the algorithm lacks enough information for an accurate estimate. In the extreme case of $k = 1$, then the algorithm simply uses only the closest profiling RSSI vector for its estimate. On the other hand, if k is too large, then the algorithm may be forced to use distant neighbours in the average calculation, introducing additional error into to the estimate. Thus, a balance needs to be struck.

To find a good value of k, we work empirically using full resolution (not

quantized) datasets. The "best" value is chosen using an exhaustive search, from $k = 1$ to the value where all possible neighbours are chosen. It is also likely that a k value that works for one k–NN algorithm might not work as well for another. Yet, to master the complexity behind the optimization of the quantization we would like to fix a “universal” k. We are therefore going to come up with a single k that is a good compromise across k–NN localization algorithms and will be used in subsequent chapers.

We can evaluate the localization performance based on its performance across a set (an ensemble) of tag measurements for which we know their ground truth location and hence can estimate their average error magnitude, $E(x)$. Similarly we can derive the standard deviation statistics, $\sigma(x)$, of the ensemble error. Rather than rely solely on $E(x)$ we will use a metric (fitness metric) which combines both $E(x)$ and $\sigma(x)$. The fitness function is as follows:

$$F = \frac{1}{E(x^2)} = \frac{1}{[E(x)]^2 + [\sigma(x)]^2} \tag{3.1}$$

This is the same fitness function used by [7]. It is also the fitness function used for evaluating the results based on GA and Tabu search in following chapters.

The results of this search for the carpark are seen in Figures 3.4 and 3.5. Localization error throughout this thesis is presented in meters. Note that there is no data for $k = 1$ for LEMON, as the algorithm requires at least $k = 2$ to function. The results are based on localization using the carpark dataset without any car present, for both the fingerprint database and the tag measurements we wish to localize. Here, we found that $k = 4$ marked a good compromise across RADAR, LANDMARC, and LEMON. In addition, we also examine $k = 5$ for LEMON, since LEMON is a special case where the weight of the furthest neighbour is zero when computing the weighted average, so essentially it appears as if $k - 1$ neighbours are used.

We then compare the choice of $k = 4$ to other choices of k that would have

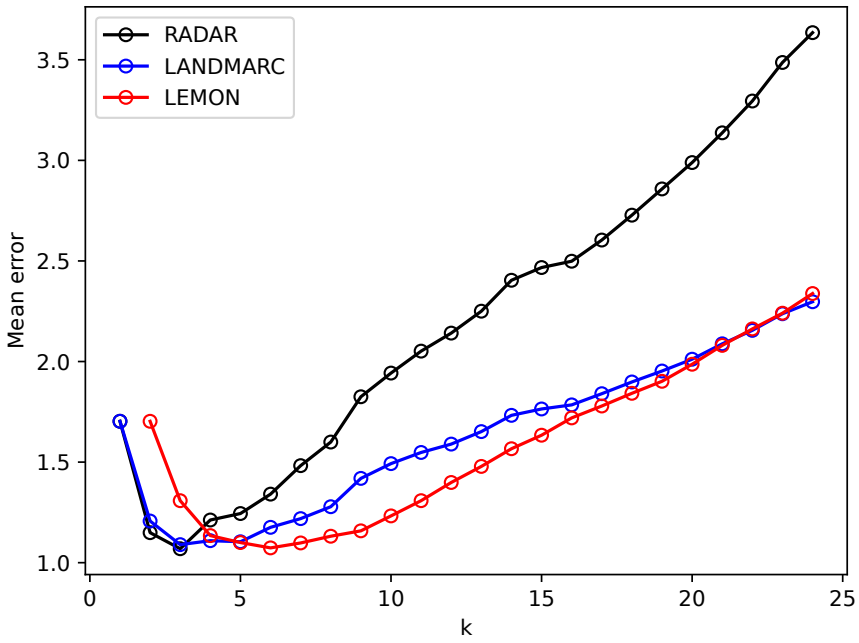


Figure 3.4: $E(x)$ vs. k (carpark dataset, no quantization).

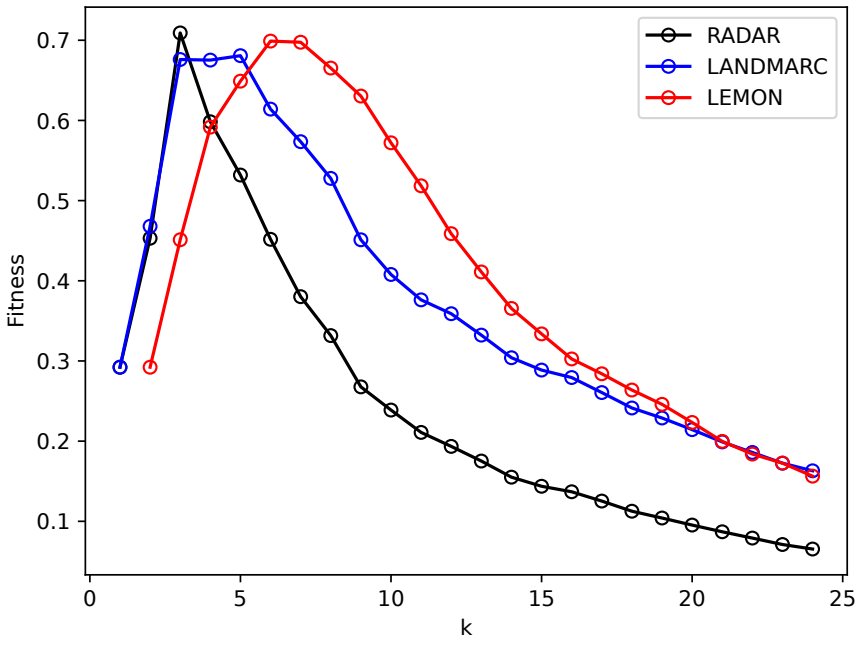


Figure 3.5: F vs. k (carpark dataset, no quantization).

Scheme	**Layout**	k	$E(x)$	$\sigma(x)$	F
RADAR	No car (NC)	3	1.07	0.52	0.71
		4	1.21	0.45	0.60
	Left (CL)	4	1.01	0.62	0.70
	Middle (CM)	1	1.49	0.35	0.43
		4	1.54	0.87	0.32
	Right (CR)	3	1.17	0.62	0.57
		4	1.24	0.61	0.52
	All	4	1.24	0.66	0.51
LANDMARC	No car (NC)	4	1.11	0.50	0.68
		5	1.10	0.50	0.68
	Left (CL)	4	0.90	0.61	0.84
	Middle (CM)	1	1.49	0.35	0.43
		4	1.40	0.83	0.38
	Right (CR)	4	1.15	0.58	0.61
		5	1.16	0.50	0.63
	All	4	1.26	0.64	0.50
LEMON	No car (NC)	4	1.13	0.64	0.59
		5	1.10	0.58	0.65
		6	1.07	0.53	0.70
	Left (CL)	4	1.01	0.71	0.66
		5	0.94	0.72	0.71
		7	0.94	0.56	0.83
	Middle (CM)	4	1.27	0.78	0.45
		5	1.31	0.79	0.43
	Right (CR)	4	1.13	0.53	0.65
		5	1.08	0.54	0.68
		6	1.09	0.46	0.71
	All	4	1.44	0.72	0.38
		5	1.38	0.71	0.42
		6	1.30	0.71	0.45

Table 3.2: Baseline localization results (carpark dataset, no quantization).

resulted in better fitness F or better $E(x)$ to ascertain whether there are other values of k that should, exceptionally explored as well. With the exception of LEMON for $k = 5$ (making it effectively $k = 4$), the results in Table 3.2 do not show any notable advantage to other values of k across the board, i.e., across all subsets of the carpark dataset. The purpose of the table it so show how $k = 4$ would perform if the set used is coming from the modified environment. For example, "Middle" means that both the fingerprint and the tags to be localized are coming from the subset of the dataset that were collected with the car in the middle of the space. Note that Table 3.2 introduces also the acronyms NC, CL, CM, and CR, we will subsequently use to refer to the presence or not, and the location, of the car obstacle. The layout "All" includes all the four subsets in one big dataset used for both the fingerprint database and for the localization of the tags. As such, it represents the most "tainted" dataset – one where presumably measurements over the same space over periods it was modified are placed in the same database and not distinguished and then the evaluation of the localization happens in a similarly mixed manner.

Coincidentally, $k = 4$ was also found to provide good results for the library dataset. The library dataset is much more sparse in comparison, with most of the data points containing just one received signal. This suggests that the best value of k should be quite small anyway. The results for the library dataset are detailed in Table 3.3, In the library dataset, we see a similar trend, with LANDMARC and LEMON achieving slightly lower mean errors. However, unlike the carpark dataset, the standard deviation of error is larger but more consistent among the localization schemes tested – the standard deviation remains roughly 1.30m for each algorithm.

3.5 Summary

This section summarizes the results under no quantization. This provides a baseline for how well each dataset can be localized using the full-resolution data. We expect that once the tag RSSI vectors are quantized, the accuracy of

k	**Scheme**	$E(x)$	$\sigma(x)$
1	RADAR	1.90	1.58
	LANDMARC	1.90	1.58
	LEMON	N/A	N/A
2	RADAR	1.93	1.31
	LANDMARC	1.88	1.41
	LEMON	1.90	1.58
3	RADAR	1.96	1.31
	LANDMARC	1.87	1.40
	LEMON	1.82	1.36
4	RADAR	2.00	1.30
	LANDMARC	1.88	1.36
	LEMON	1.82	1.31
5	RADAR	2.07	1.35
	LANDMARC	1.87	1.34
	LEMON	1.81	1.26
6	RADAR	2.20	1.46
	LANDMARC	1.87	1.33
	LEMON	1.80	1.22
7	RADAR	2.29	1.59
	LANDMARC	1.88	1.32
	LEMON	1.80	1.19

Table 3.3: Baseline localization results (library dataset, no quantization).

their localization will be reduced.

For each of the following datasets (the carpark and the library), three k-NN algorithms are applied (RADAR, LANDMARC and LEMON), using the value of $k = 4$ for each algorithm, while examining also $k = 5$ for LEMON. In every case, full-resolution data is used for both the fingerprint database and the tag measurements to be localized. We summarize the reference performance for the carpark dataset in Table 3.4 and for the library dataset in Table 3.5. They will form the basis of comparisons in subsequent chapters.

Scheme	$E(x)$ ($\sigma(x)$)
RADAR	1.21 (0.45)
LANDMARC	1.11 (0.50)
LEMON	1.13 (0.64)
LEMON ($k = 5$)	1.10 (0.58)

Table 3.4: Reference performance for $k = 4$ (carpark dataset, no quantization).

Scheme	$E(x)$ ($\sigma(x)$)
RADAR	2.00 (1.30)
LANDMARC	1.88 (1.36)
LEMON	1.82 (1.31)
LEMON ($k = 5$)	1.81 (1.26)

Table 3.5: Reference performance for $k = 4$ (library dataset, no quantization).

Chapter 4

1-bit Quantization

In this chapter, we explore the simplest case involving quantization: 1-bit quantization. If we take RSSI to be a proxy for physical distance, then this special case treats each peg as if it was a proximity sensor. That is, if the signal strength is higher than a certain value, then we consider that the tag is "near" to the tag. Otherwise, it is “far” and doesn’t "trigger" the proximity sensor. As there are only 2 possible values for quantized RSSI, each fingerprint can be expressed in 2^n bits, assuming n pegs.

The main question, then, is what the threshold/boundary value should be to separate the “near” from the “far” readings. In this chapter, we first consider the case where every peg shares the same threshold (global threshold). Subsequently, we expand the problem to consider the case where every peg has its own threshold value (local thresholds). In both cases, we compare the localization accuracy of 1-bit quantization across the three k–NN based algorithms when 1-bit quantization is applied, and when no quantization is used.

4.1 Methodology

We consider two types of thresholds: global and local. For global thresholds, the same value of r_{thr} is used for the RSSI values received from any peg. With local thresholds, the RSSI values from each peg are quantized in a different way, using a different r_{thr} for each. If there are n pegs, there are n threshold values, $r_{thr_1}, r_{thr_2}, ..., r_{thr_n}$.

For global thresholds, finding r_{thr} is relatively trivial. RSSI values at full resolution are typically 7 or 8 bits, so there are at most $2^8 = 256$ values through which to search and this is done exhaustively. Local thresholds have an added layer of complexity. If there are m pegs, then the search space is now 2^{7m} or 2^{8m} wide, which necessitates more sophisticated search techniques. Here, we use the elitist genetic algorithm (EGA) with tournament selection. The EGA algorithm is described in Section 2.4.1 and the runtime parameters used are summarized in Table 4.1.

Parameter	**Value**
Population size (chromosomes)	500
Number of generations	10000
Crossover probability	0.90
Mutation probability	0.15

Table 4.1: EGA parameters.

In evaluating these quantization thresholds, we compute the localization of the set of RSSI samples after quantizing. The computed locations are compared to their known locations using the Euclidean distance between the two points. This distance is referred to as the "error" for that particular sample. Across all samples in a dataset, the average of all sample errors is reported as mean error. The sample standard deviation is also calculated. The effectiveness of these quantization thresholds is evaluated using the fitness function, which takes into account both the mean error and standard deviation of the error (Equation 3.1).

We will need to represent these "near" and "far" values with a representative RSSI value, as the k-NN algorithms require actual RSSI values to estimate the location of a tag. Suppose the collected RSSI values range from r_{min} to r_{max} with r_{thr} lying between the two. This creates two RSSI ranges, one from r_{min} to r_{thr} and another from r_{thr} to r_{max}.We use the midpoint of each RSSI range as the representative RSSI value of the corresponding bin. Concretely, $R_0 = r_{min} + \frac{r_{thr}+r_{min}}{2}$ and $R_1 = r_{thr} + \frac{r_{thr}+r_{max}}{2}$ are the representa-

tive values for the “far” and “near” (0 and 1 respectively). For example, if the RSSI vector[1] for a given tag is (131, 104, 88), and $r_{min} = 50$, $r_{max} = 150$, and $r_{thr} = 100$, then the quantized form of that vector is (125, 125, 75) which is also representable as (1, 1, 0). Note that the fingerprint database still contains the non-quantized RSSI values and only the RSSI values for the tag RSSI vectors that need to be localized will be quantized.

Another detail to note is that even in the local thresholds case, the global r_{min} and r_{max} values are common across all pegs and corresponding to the minimum and maximum recorded RSSI value present in the fingerprint database. Only the threshold value, r_{thr_i} differs across pegs.

In the results for this chapter, we will compare the localization error $E(x)$ and $\sigma(x)$ across: no quantization, "global 1-bit" quantization, and "local 1-bit" quantization, each for a number of configurations of the carpark dataset, as well as for the library dataset. As quantization inherently reduces the quantity of data available to perform localization, we expect that error and standard deviation will be larger for the quantized configurations compared to the results for no quantization. For the same reason, we expect that the localization error for "local 1-bit" quantization will be lower than that of "global 1-bit" quantization. The hope is that by having a local threshold for each peg, any differences between pegs – whether they are differences in environment, configuration, or equipment – will be mitigated.

4.1.1 High-Level Flowchart

Figure 4.1 presents the overall process for searching for quantization thresholds (solid red arrows), as well as their subsequent testing (dashed green arrows). In all cases, the localization scheme chosen is treated as a black box parameterized by k. There is a distinction of a “training” set, and a “testing” set. The training set is the set of the fingerprint measurements as well as

[1]As explained in the related work chapter, the choice of units for representing RSSI value is of no consequence to the definition of the k neighbors. Here the values used are in units used during the RSSI collection with the CC1100 chipset for the carpark dataset.

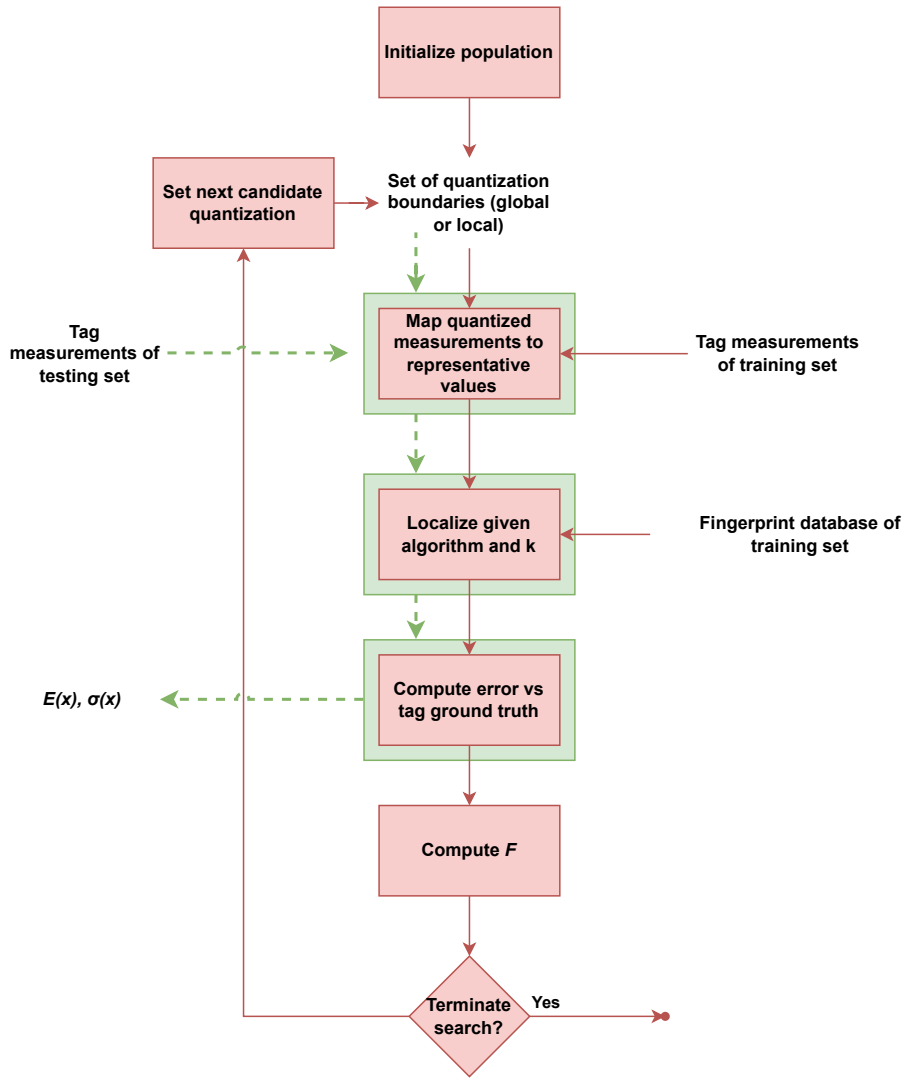


Figure 4.1: Flowchart of the optimal quantization search and subsequent testing.

the RSSI tag measurements collected under the same conditions as the fingerprint measurements. The latter are used in the process of determining the optimal quantization boundaries. The testing set is a set of RSSI measurements for tags to be localized that may have been collected under entirely different conditions. For example, one can train on the NC dataset and test in the CR dataset. The process outlined in *red* is involved in the optimal quantization search process, regardless of whether it is GA or, in the next chapter, Tabu search. Once a good set of quantization boundaries is found, it can be evaluated using the testing data set following the *green* path in the flowchart.

Intuitively, one can think of the separation between training and testing as an opportunity to expore whether quantization determined from an original set of fingerprints can be reliably used to localize points in the same space, after a change in the environment has occured (like the presence of the car in the case of our carpark dataset).

4.2 Results

4.2.1 Carpark Dataset

The carpark dataset is not large. For instance, in the most complete arrangement, that with no car, there are only 496 data points across 25 positions in the training set.

As our methodology requires this dataset to also train for the thresholds, this leads to the situation where it is very difficult to split these 96 data points between a training and an evaluation dataset. Therefore, where the car arrangement is the same for both the training and testing (i.e., not testing for a "change in environment"), the same tag data points are also used to both in the training and in testing, which could produce favorable results; the real error is likely worse than reported. However, we also examine cases where the quantization thresholds are tested using a different environment than the training environment. For example, after obtaining the optimal set of quantization thresholds for an environment with no car (NC), the resulting quantization can be tested using the tag measurements in the environment with a car placed in the middle (CM) of the space. In this fashion, we are able to capture a change in environment between the training phase and when the quantized localization system is used. Our results are summarized in Tables 4.3 and 4.4, where the "train" and the "test" refer to the separate training (optimization) and testing setting. The results in the non-quantized case are also brought in from Table 4.2.

A general observation is that, compared to no quantization, using global 1-

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** ($k = 5$)
NC	NC	1.21 (0.45)	1.11 (0.50)	1.14 (0.64)	1.10 (0.58)
	CL	0.99 (0.54)	0.92 (0.53)	0.97 (0.63)	0.91 (0.58)
	CM	1.42 (0.91)	1.42 (0.94)	1.66 (1.22)	1.51 (1.08)
	CR	1.31 (0.59)	1.20 (0.59)	1.21 (0.62)	1.15 (0.56)
CL	NC	1.14 (0.59)	0.95 (0.58)	0.99 (0.68)	0.93 (0.61)
	CL	1.01 (0.62)	0.90 (0.61)	1.01 (0.71)	0.94 (0.72)
	CM	1.44 (0.94)	1.36 (0.99)	1.47 (1.13)	1.38 (1.06)
	CR	1.46 (0.71)	1.33 (0.71)	1.41 (0.97)	1.31 (0.84)
CM	NC	1.34 (0.85)	1.25 (0.80)	1.18 (0.78)	1.20 (0.77)
	CL	1.18 (0.55)	1.15 (0.56)	1.36 (0.69)	1.26 (0.63)
	CM	1.54 (0.87)	1.40 (0.83)	1.27 (0.77)	1.31 (0.79)
	CR	1.29 (0.80)	1.26 (0.72)	1.30 (0.66)	1.28 (0.63)
CR	NC	1.36 (0.89)	1.25 (0.86)	1.23 (0.75)	1.15 (0.77)
	CL	1.37 (0.78)	1.29 (0.84)	1.44 (0.97)	1.34 (0.97)
	CM	1.79 (1.17)	1.67 (1.20)	1.57 (1.10)	1.54 (1.16)
	CR	1.24 (0.61)	1.14 (0.58)	1.13 (0.53)	1.08 (0.54)

Table 4.2: Localization performance (carpark dataset, no quantization).

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** ($k = 5$)
NC	NC	1.25 (0.66)	1.23 (0.66)	1.44 (0.72)	1.36 (0.68)
	CL	1.48 (0.63)	1.44 (0.66)	1.67 (0.83)	1.56 (0.74)
	CM	1.76 (1.02)	1.74 (1.07)	2.22 (1.37)	2.02 (1.31)
	CR	1.67 (0.89)	1.63 (0.89)	1.71 (0.95)	1.63 (0.92)
CL	NC	1.62 (0.73)	1.53 (0.71)	1.74 (1.27)	1.71 (1.21)
	CL	1.47 (0.78)	1.41 (0.72)	1.41 (0.70)	1.40 (0.70)
	CM	1.96 (1.22)	1.96 (1.27)	2.05 (1.24)	1.99 (1.11)
	CR	1.87 (1.07)	1.88 (1.07)	2.12 (1.37)	2.05 (1.29)
CM	NC	1.58 (0.83)	1.71 (0.98)	1.71 (1.07)	1.69 (1.05)
	CL	1.72 (0.76)	1.62 (0.84)	1.63 (0.94)	1.63 (0.87)
	CM	1.69 (0.65)	1.67 (0.78)	1.58 (1.01)	1.60 (0.92)
	CR	1.58 (0.73)	1.62 (0.89)	1.88 (1.07)	1.74 (0.98)
CR	NC	1.43 (0.86)	1.45 (0.88)	1.84 (1.03)	1.65 (0.95)
	CL	1.67 (0.88)	1.77 (0.97)	2.08 (1.25)	2.00 (1.11)
	CM	1.79 (1.37)	1.84 (1.38)	2.06 (1.26)	1.93 (1.45)
	CR	1.49 (0.87)	1.50 (0.88)	1.95 (0.81)	1.66 (0.95)

Table 4.3: Localization performance (carpark dataset, global 1-bit quantization).

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** ($k = 5$)
NC	NC	0.96 (0.47)	0.95 (0.48)	0.92 (0.48)	0.84 (0.44)
	CL	1.74 (0.76)	1.65 (0.70)	1.57 (0.97)	1.79 (1.28)
	CM	1.90 (1.12)	1.68 (1.05)	2.16 (1.28)	1.90 (1.28)
	CR	1.79 (0.95)	1.89 (0.90)	1.93 (0.94)	1.84 (0.88)
CL	NC	1.91 (0.93)	1.77 (0.88)	1.54 (1.05)	1.66 (0.88)
	CL	0.89 (0.45)	0.92 (0.45)	0.97 (0.52)	1.00 (0.44)
	CM	2.00 (1.52)	1.94 (1.34)	1.82 (1.10)	1.77 (1.11)
	CR	2.36 (1.93)	1.96 (1.47)	2.39 (1.54)	1.96 (1.34)
CM	NC	1.85 (0.88)	1.70 (1.30)	2.11 (1.50)	1.75 (0.93)
	CL	1.82 (1.13)	1.98 (0.84)	1.86 (1.11)	1.01 (1.04)
	CM	1.01 (0.70)	1.06 (0.66)	1.16 (0.54)	1.13 (0.52)
	CR	2.05 (1.26)	1.76 (0.95)	2.20 (0.93)	2.25 (1.55)
CR	NC	2.04 (1.04)	1.85 (1.11)	2.17 (1.39)	1.75 (0.93)
	CL	2.26 (1.05)	1.85 (0.89)	2.06 (1.40)	1.85 (1.08)
	CM	1.86 (1.20)	1.80 (1.33)	2.30 (1.69)	2.01 (1.23)
	CR	1.00 (0.42)	1.06 (0.49)	0.96 (0.58)	0.99 (0.50)

Table 4.4: Localization performance (carpark dataset, local 1-bit quantization).

bit quantization with global thresholds typically results in slightly increased errors in localization. For no quantization, while $E(x)$ is rather low (generally 1 to 1.5m dependent on arrangement), $\sigma(x)$ can be rather high (generally in the 0.5m to 1.0m range). When transitioning to global 1-bit quantization, we see an across the board increase for both $E(x)$ and $\sigma(x)$. Note that the fitness function 3.1, F, is used to guide the search towards reducing both $E(x)$ and $\sigma(x)$. It is therefore entirely possible to decrease $E(x)$ further, but at the cost of increasing $\sigma(x)$. The goal of the fitness function is to balance both. The boundary values for each localization algorithm were found to be almost identical. Hence the performance differences stem from the individual logic of each k–NN scheme rather than a different optimal quantization.

Comparing the local 1-bit quantization to the no quantization state, the picture becomes more complicated. In some cases, $E(x)$ is marginally higher than without quantization, but in other cases, it can be even lower than without quantization! This trend continues across both $E(x)$ and $\sigma(x)$.

When quantization is applied, $E(x)$ generally increases along with $\sigma(x)$. However, this is not the case across the board. For example, in the NC/NC configuration for RADAR, $E(x)$ is actually lower for local 1-bit quantization than without quantization. This is perhaps due to the non-weighted nature of RADAR. With $k = 4$ neighbouring fingerprints, each neighbour is given the same weight when comparing the average location. However, a particular neighbour may not be of the same "quality" as its other neighbours and skew the average away from the actual location. Local quantization may suppress that effect by injecting peg-dependent information in the quantization process. For instance, if a particular peg's transmissions are stronger or weaker than another, the quantization boundaries applicable to the individual peg are accordingly influenced.

Where the training and testing set differ, we see a marked increase in $E(x)$. The optimal boundaries found are suited for a particular environment, and once this environment changes, these values are no longer ideal. For instance, in the case of local 1-bit quantization, in the NC environment, we witness an increase in $E(x)$ which is close to doubling of the error when we test with the car to the right (CR). With global 1-bit quantization, the corresponding (train in NC; test in CR) increase of $E(x)$ is proportionately modest, albeit it starts from an already inflated $E(x)$. Generally, as the environment changes, local quantization performs worse than global quantization, and in each case, localization accuracy in the changed environment is significantly worse than under no quantization.

We find that quantization, whether global or local 1-bit quantization, removes many of the performance differences between the three localization schemes. While there might have been very slight differences without quantization, these differences do not exactly transfer when quantization is applied. For instance, for the NC/NC configuration, RADAR performs slightly worse than LEMON does without quantization. However, when either global or local 1-bit quatnization is applied, RADAR pulls ahead.

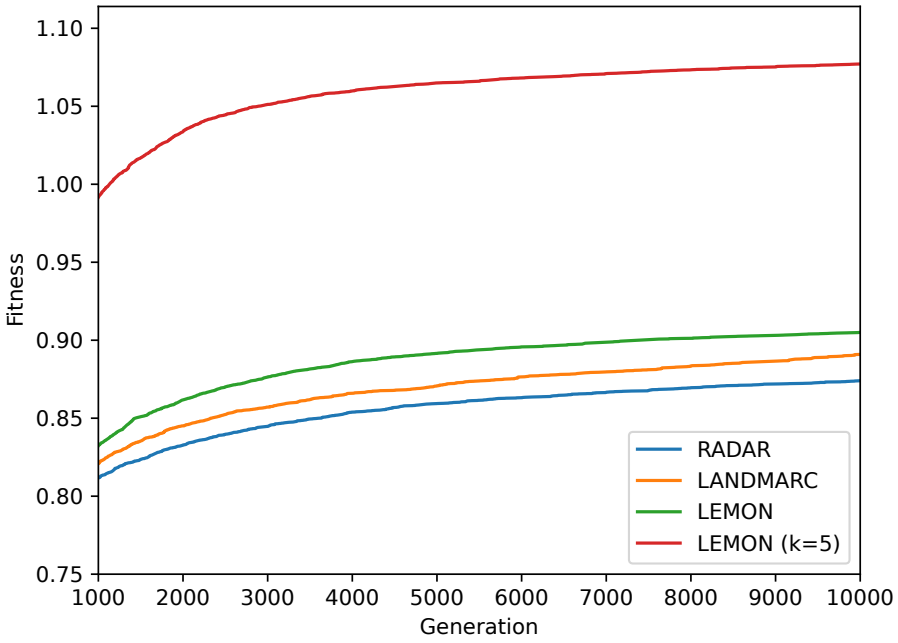


Figure 4.2: F vs. GA generation (carpark NC dataset, local 1-bit quantization).

Figure 4.2 shows the fitness curve for local 1-bit quantization using EGA search. Note that the graph starts at generation 1000, as prior to this all four schemes rapidly rise from a fitness of roughly 0.3 up to near the peak level. Each curve consists of the mean fitness of 250 runs at the stated generation. From here, we can see that fitness generally rapidly increases up to around the 2000 generation mark, but quickly plateaus. The choice of localization algorithm (RADAR vs LANDMARC) appears to have little effect on this rate.

4.2.2 Library Dataset

The library dataset, containing 105 data points in the fingerprint database and 419 points to be localized, is for a space that is much more spread out than the carpark. As seen in Table 3.1, for most locations, transmissions are received from only one beacon. Only 7% of the RSSI vectors in the dataset contain transmission strengths from more than two beacons (out of 13 beacons in total). Therefore, almost all the the RSSI vectors in the dataset are

Scheme	$E(x)$ ($\sigma(x)$)		
	no quantization	**global 1-bit**	**local 1-bit**
RADAR	1.99 (1.30)	2.08 (1.27)	1.94 (1.21)
LANDMARC	1.89 (1.36)	2.02 (1.17)	1.91 (1.17)
LEMON	1.82 (1.31)	2.04 (1.23)	2.27 (1.63)
LEMON ($k = 5$)	1.81 (1.26)	1.94 (1.17)	1.86 (1.21)

Table 4.5: Localization performance (library dataset, 1-bit quantization)

incomplete. That is, there are many missing values in each RSSI vector. Each k-NN algorithm requires a numerically complete RSSI vector, without any missing values. This is because we need to know the distance in signal space between any two samples (see (2.2)). Thus, we will need an imputed value for these missing values. As discussed in Section 3.2.1, we chose a value of -120 dBm to be imputed when no measurement from a peg is present.

A major concern with any particular imputed value is how much separation there is between the imputed value and "real" recorded values. As described in Chapter 3, the RSSI values for the iBeacons ranged between -88 and -55 dBm (with most of the values in the -80 to -75 dBm range). The imputed value of -120 dBm is different enough from the real values and that contributes to lessening the chances that, a peg that is present in one vector and imputed in another vector, appear to be near each other.

As seen in Table 4.5, the trends are similar to that for the carpark data. Namely, there is little difference between any of the schemes. More interestingly, there is virtually no additional error caused by quantization. This is very likely linked to the sparse nature of the dataset. The majority of the RSSI vectors contain just one “heard” peg, with all the rest being imputed values and its nearest neighbors are likely to be all that hear also just the same peg. In other words, the exact RSSI value from the heard peg is of little consequence.

RADAR, because of $k = 4$, is often forced to pick vectors with “not-heard” (imputed) values as part of its k neighbours, despite these vectors expressing

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** ($k = 5$)
NC	NC	1.58 (0.89)	1.56 (0.87)	1.74 (1.04)	1.67 (0.90)
	CL	1.84 (1.00)	1.80 (1.02)	1.93 (1.27)	1.81 (1.13)
	CM	2.15 (1.16)	2.15 (1.20)	2.40 (1.61)	2.30 (1.48)
	CR	1.70 (0.92)	1.69 (0.94)	1.96 (1.04)	1.85 (1.03)
CL	NC	1.63 (0.85)	1.56 (0.87)	1.69 (1.09)	1.61 (0.98)
	CL	1.70 (0.84)	1.67 (0.83)	1.93 (0.99)	1.80 (0.92)
	CM	1.99 (1.07)	2.03 (1.10)	2.41 (1.39)	2 .27 (1.34)
	CR	1.88 (1.13)	1.91 (1.14)	2.33 (1.34)	2.12 (1.30)
CM	NC	1.73 (0.76)	1.66 (0.81)	1.75 (1.04)	1.68 (0.90)
	CL	1.98 (0.95)	2.01 (0.97)	2.36 (1.23)	2.23 (1.17)
	CM	2.30 (1.29)	2.33 (1.33)	2.65 (1.68)	2.55 (1.62)
	CR	1.73 (1.11)	1.71 (1.09)	1.88 (0.98)	1.77 (1.01)
CR	NC	1.53 (0.88)	1.53 (0.88)	1.91 (1.06)	1.69 (0.91)
	CL	1.90 (0.94)	1.90 (0.90)	2.15 (1.23)	2.01 (1.03)
	CM	2.25 (1.24)	2.28 (1.28)	2.57 (1.55)	2.48 (1.46)
	CR	1.71 (0.80)	1.72 (0.79)	2.15 (1.19)	1.97 (0.99)

Table 4.6: Localization performance (carpark dataset, linear 1-bit quantization)

locations that are being potentially very far away. Since RADAR does not use a weighted average when computing location, under quantization, those distant in physical space but neighbouring in signal space neighbors will be treated the same as the closer ones.

4.2.3 Fixed Quantization

We also test localization accuracy using the fixed (i.e., not the product of a search process) localization boundaries described in equations 2.6–2.9. These equations are called the linear, square-root, cube-root, and log quantization respectively. For these boundaries, only the minimum and maximum RSSI values are used to compute these boundaries. However, using fixed boundaries allows us a glimpse into quantization without any kind of search for the best boundaries. Thus, we expect that localization accuracy will be worse when using fixed boundaries as opposed to boundaries we search for. After all, these fixed boundaries are included in the search space, so a search can

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** ($k = 5$)
NC	NC	1.87 (1.14)	1.79 (1.14)	1.84 (1.11)	1.76 (1.09)
	CL	1.41 (0.68)	1.39 (0.67)	1.70 (0.84)	1.60 (0.73)
	CM	1.75 (1.03)	1.73 (1.07)	2.22 (1.37)	2.01 (1.32)
	CR	1.69 (0.93)	1.66 (0.92)	1.75 (1.01)	1.66 (0.97)
CL	NC	1.78 (1.20)	1.79 (1.19)	1.83 (1.10)	1.84 (1.08)
	CL	1.70 (0.79)	1.58 (0.73)	1.47 (0.78)	1.46 (0.74)
	CM	1.88 (1.10)	1.87 (1.15)	2.06 (1.45)	2.01 (1.35)
	CR	1.97 (0.99)	1.98 (0.99)	2.18 (1.27)	2.11 (1.13)
CM	NC	1.59 (0.88)	1.57 (0.89)	1.64 (1.00)	1.64 (0.97)
	CL	1.57 (0.69)	1.57 (0.63)	1.74 (0.87)	1.66 (0.69)
	CM	1.78 (0.99)	1.79 (1.01)	1.95 (1.21)	1.93 (1.11)
	CR	1.65 (0.96)	1.67 (0.89)	1.80 (0.86)	1.71 (0.80)
CR	NC	2.07 (1.43)	2.00 (1.42)	1.99 (1.41)	1.96 (1.38)
	CL	1.79 (0.83)	1.89 (1.94)	2.22 (1.17)	2.13 (1.09)
	CM	1.69 (1.33)	1.73 (1.32)	2.17 (1.58)	1.90 (1.39)
	CR	1.54 (0.90)	1.54 (0.90)	1.88 (1.19)	1.69 (0.97)

Table 4.7: Localization performance (carpark dataset, $\sqrt[2]{}$ 1-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** ($k = 5$)
NC	NC	1.74 (1.11)	1.70 (1.16)	1.86 (1.36)	1.74 (1.24)
	CL	1.55 (0.68)	1.52 (0.71)	1.73 (0.87)	1.603 (0.81)
	CM	2.02 (1.18)	2.01 (1.21)	2.27 (1.43)	2.17 (1.38)
	CR	2.03 (1.11)	2.00 (1.12)	2.04 (1.11)	1.92 (1.08)
CL	NC	1.85 (1.17)	1.77 (1.23)	1.89 (1.37)	1.87 (1.31)
	CL	1.53 (0.79)	1.47 (0.74)	1.57 (0.81)	1.49 (0.77)
	CM	2.13 (1.33)	2.14 (1.36)	2.27 (1.68)	2.24 (1.58)
	CR	2.20 (1.11)	2.20 (1.11)	2.37 (1.28)	2.26 (1.22)
CM	NC	1.92 (1.44)	1.86 (1.44)	1.89 (1.44)	1.89 (1.40)
	CL	1.47 (0.75)	1.48 (0.72)	1.72 (0.86)	1.61 (0.82)
	CM	2.07 (1.16)	2.06 (1.18)	2.17 (1.28)	2.16 (1.24)
	CR	1.98 (1.03)	1.96 (0.96)	1.99 (1.02)	1.94 (0.95)
CR	NC	1.85 (1.25)	1.83 (1.27)	2.10 (1.42)	1.98 (1.37)
	CL	1.67 (0.91)	1.79 (0.98)	2.28 (1.20)	2.11 (1.10)
	CM	2.06 (1.43)	2.07 (01.43)	2.30 (1.60)	2.23 (1.46)
	CR	1.78 (0.99)	1.82 (0.98)	2.13 (1.26)	1.92 (1.12)

Table 4.8: Localization performance (carpark dataset, $\sqrt[3]{}$ 1-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** ($k = 5$)
NC	NC	1.62 (0.79)	1.52 (0.80)	1.50 (0.81)	1.46 (0.80)
	CL	1.74 (0.96)	1.65 (0.98)	1.55 (0.96)	1.58 (0.97)
	CM	1.70 (0.89)	1.67 (0.98)	1.92 (1.23)	1.78 (1.09)
	CR	1.78 (1.01)	1.74 (1.05)	1.97 (1.09)	1.89 (1.02)
CL	NC	1.75 (0.98)	1.73 (1.05)	1.90 (1.33)	1.80 (1.26)
	CL	1.78 (1.10)	1.72 (1.14)	1.80 (1.34)	1.82 (1.31)
	CM	1.94 (0.86)	1.87 (0.95)	2.05 (1.37)	2.00 (1.24)
	CR	1.96 (1.04)	1.96 (1.09)	2.25 (1.48)	2.14 (1.39)
CM	NC	1.67 (0.82)	1.65 (0.82)	1.73 (0.93)	1.71 (0.89)
	CL	1.65 (0.83)	1.62 (0.86)	1.66 (0.97)	1.65 (0.91)
	CM	1.75 (0.91)	1.76 (0.97)	1.91 (1.14)	1.84 (1.06)
	CR	1.80 (0.92)	1.85 (0.91)	2.13 (1.11)	2.03 (0.98)
CR	NC	1.77 (0.89)	1.84 (1.12)	1.84 (1.03)	1.83 (0.95)
	CL	1.99 (1.08)	2.04 (1.31)	1.89 (1.35)	1.89 (1.30)
	CM	1.74 (1.04)	1.72 (1.07)	1.91 (1.28)	1.78 (1.08)
	CR	1.90 (1.08)	1.97 (1.08)	2.33 (1.22)	2.15 (1.13)

Table 4.9: Localization performance (carpark dataset, log 1-bit quantization)

only improve further what the fixed ones can achieve.

With 1 bit, the equations yield only a single value, which we use as a global threshold for each of the pegs. We then quantize all data using the same method, and localize them using our three k-NN localization schemes. The localization results are summarized in Tables 4.6–4.9. Generally, the localization accuracy using fixed boundaries with 1 bit is poor, with $E(x)$ being roughly double what we obtain using optimal global 1-bit thresholds.

4.3 Coverage

After quantization, the number of distinct RSSI vectors is drastically reduced. In the case of 1-bit quantization, there are only 2^n possible vectors for n pegs. With the carpark dataset with 10 pegs, this means that there are only 512 possible vectors. This corresponds to 512 locations that can be produced as output of the localization algorithm. Moreover, not all 512 locations are necessarily distinct.

As noted in Section 2.1.8, the possible localization points fall within the convex hull of the profiling points, and oftentimes (as in our case) in the convex hull of the pegs. The carpark layout shown in Figure 3.1 has pegs deployed on the edges but not at the corners, hence there are four triangular areas at the four corners where no localization is possible using any of the presented k–NN schemes. The total area of $4(2m^2) = 8m^2$ of those four corner triangles represents a $8/(8 \times 12) = 8.33\%$ of the total rectangular area. Hence, the k–NN schemes can, at most, cover $100 - 8.33 = 91.67 \simeq 92\%$ of the rectangular area.

The *coverage* diagrams in Figures 4.3, 4.4, 4.5 and 4.6 plot these localization across the area for global (Figures 4.3 and 4.4) and local (Figures 4.5 and 4.6) 1-bit quantization. Furthermore Figures 4.3 and 4.5 show the quantization for training without a car present while Figures 4.4 and 4.6 show results with a car placed to the left part of the area. Placements other than in the left (so, right and center) produced similar results to those of the left placement shown in Figures 4.4 and 4.6. All Figures preserve the scale of the space used in the carpark dataset, which is 8 meters by 12 meters in size.

One way to interpret the coverage diagrams is to appreciate how complete is the coverage of the area given the quantization that has been performed. Quantifying the coverage of an area requires that we put some "weight" to the points (as points have a measure of zero). To this end, we approximate a localized point by a square of approximately $27 \times 27mm^2$ area (equivalent to the square size for the points in the coverage figures). This allows us to describe the overall coverage for a given scheme and keep it equivalent to the visual representation we have chosen. RADAR turns out to have a very uniform and sparse coverage. The 1-bit quantization schemes appear to be sparse in their coverage, as one would expect. Hence, numerically, their coverage as a percentage of the overall area is unimpressive. Even then, the average coverage of RADAR is far inferior at no more than 0.17% compared to all the other schemes that achieve a coverage between a narrow range of 0.74 and 0.79% (a more than four-fold increase in coverage). Nevertheless, the cover-

Scheme	$E(x)$ ($\sigma(x)$)			
	none	**linear**	**global**	**local**
RADAR	1.21 (0.45)	1.58 (0.89)	1.25 (0.66)	0.96 (0.47)
LANDMARC	1.11 (0.50)	1.56 (0.87)	1.23 (0.66)	0.95 (0.48)
LEMON	1.14 (0.64)	1.74 (1.04)	1.44 (0.72)	0.92 (0.48)
LEMON ($k = 5$)	1.10 (0.58)	1.67 (0.90)	1.38 (0.68)	0.84 (0.44)

Table 4.10: 1-bit quantization performance (carpark dataset, train:NC, test:NC).

age statistics become even more distinct when we present in the next chapter 2-bit quantization schemes.

Generally, RADAR has the poorest coverage. As the neighbours are used in the localization calculation in an unweighted form, many of these vectors will share the same location, as shown in Figure 4.3a. In contrast, the weighted-average algorithms carry a much higher range of points, as seen in Figure 4.3d, which shows LEMON with $k = 5$ and a 1-bit global localization scheme. Here, more distant neighbours are weighted less in the localization calculation, so a more spread out set of locations become possible as output of the localization.

Figure 4.5 shows the coverage of a 1-bit local quantization scheme does not differ notably from that with 1-bit global shown in Figure 4.3. More striking is the difference when the training was limited because of the presence of the obstacle (the car in the left) shown (for global) in Figure 4.4 and (for local) in Figure 4.6. The obstacle's presence impacts the measurements collected, especially around it, and hence, the post-quantization coverage lack dense points in the area where the obstacle is present. While LANDMARC seems to fare better when local quantization is used, there appears to be the opposite impact on LEMON which seems to further develop artifacts such co-linear (and hence less random) localized points as shown at the top areas of Figures 4.6c and 4.6d.

Scheme	$E(x)$ ($\sigma(x)$)			
	none	**linear**	**global**	**local**
RADAR	1.31 (0.59)	1.70 (0.92)	1.67 (0.89)	1.79 (0.95)
LANDMARC	1.20 (0.59)	1.69 (0.94)	1.63 (0.89)	1.89 (0.90)
LEMON	1.21 (0.62)	1.96 (1.04)	1.71 (0.95)	1.93 (0.94)
LEMON ($k = 5$)	1.15 (0.56)	1.85 (1.03)	1.63 (0.92)	1.84 (0.88)

Table 4.11: 1-bit quantization performance (carpark dataset, train:NC, test:CR).

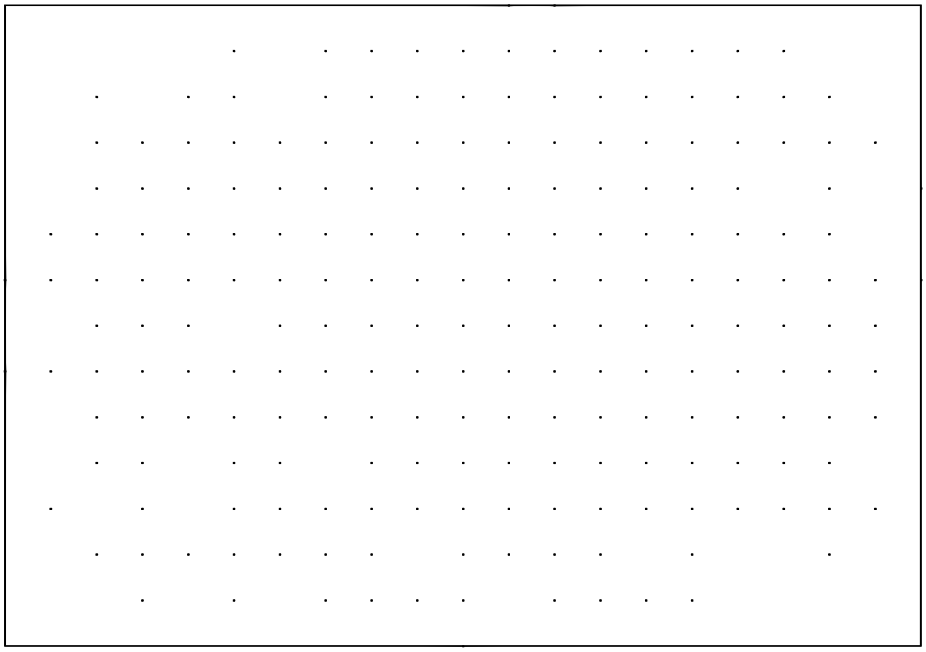
4.4 Conclusions

In both the carpark and the library dataset, we see localization error decrease from localization using full resolution data to localization using 1-bit quantized values. However, the decrease in accuracy is often minimal, particularly with the library dataset. It might be possible to improve on this by employing more than 1 bit for the quantized representation – a task we pursue in the next Chapter.

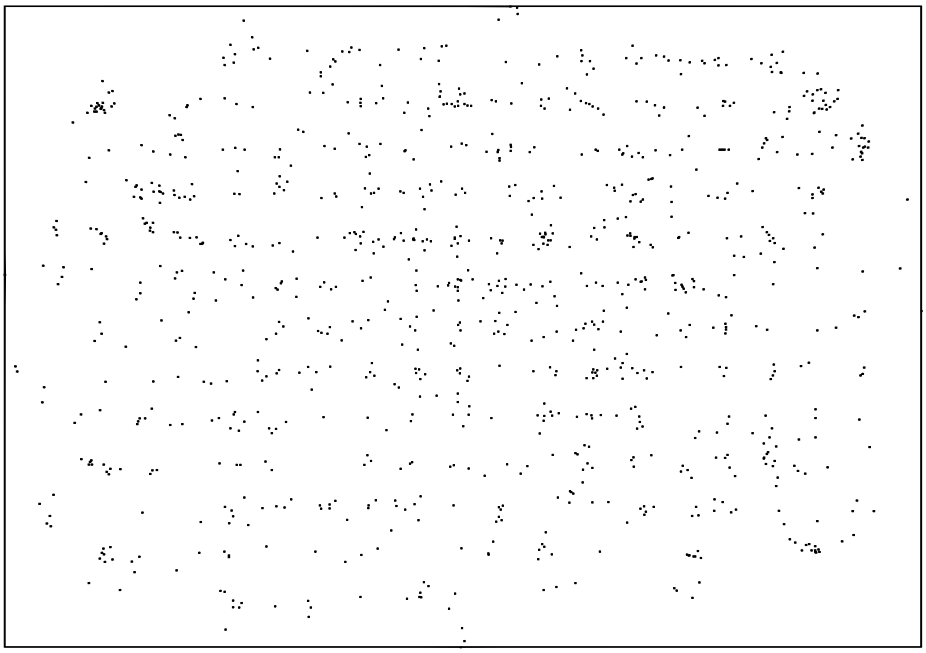
Since the library dataset involves much more sparse data, and in many cases involves just one received peg signal for a particular data point, there is little information to be lost in quantization. Thus, at least when data is sparse, quantizing down to the bare minimum, a single threshold globally turning each beacon into an on-off sensor seems adequate. This conclusion does not apply to the car park dataset in its entirety. Here, the error can increase substantially from no quantization to global 1-bit quantization since there is high coverage of the region of interest by the pegs.

With the carpark dataset, we can also see that localization accuracy is much better when using optimal quantization boundaries rather than fixed boundaries. This difference is more pronounced for the datasets involving vehicles.

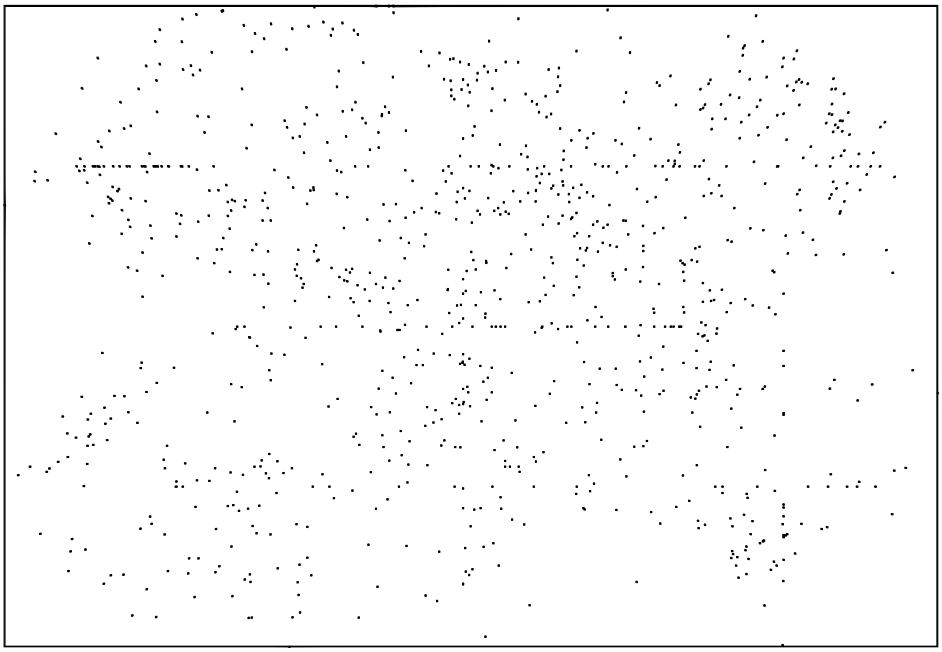
By way of summary of this chapter, consider Tables 4.10 and 4.11. Table 4.10 summarizes the results where NC is used for both training and testing, and Table 4.11 does the same for the case where NC is used for training and CR is used for testing. The fixed boundary scheme used in both cases is the linear quantizer, which separates the RSSI space into 2 equal regions.



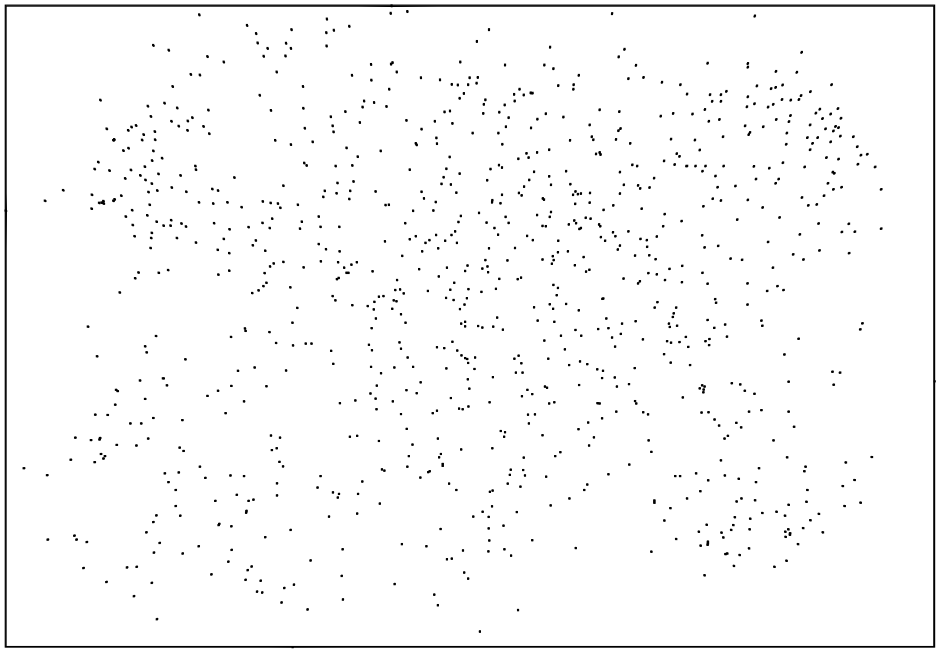
(a) RADAR



(b) LANDMARC

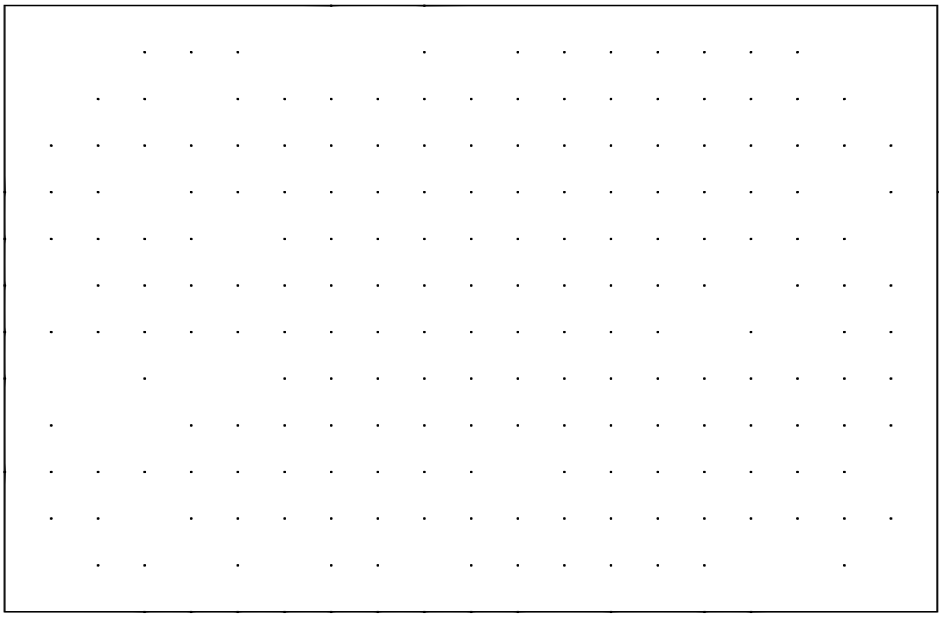


(c) LEMON

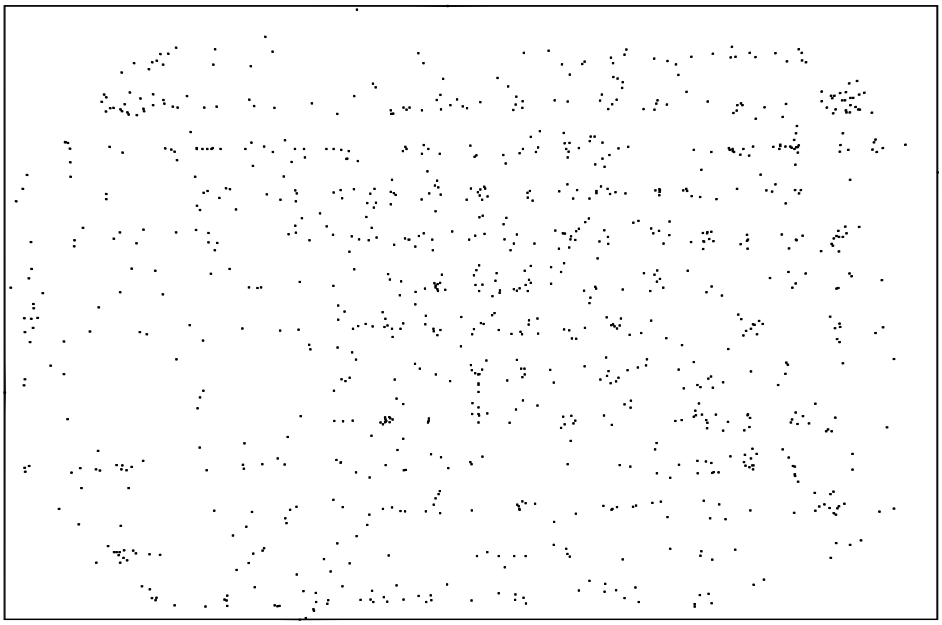


(d) LEMON ($k = 5$)

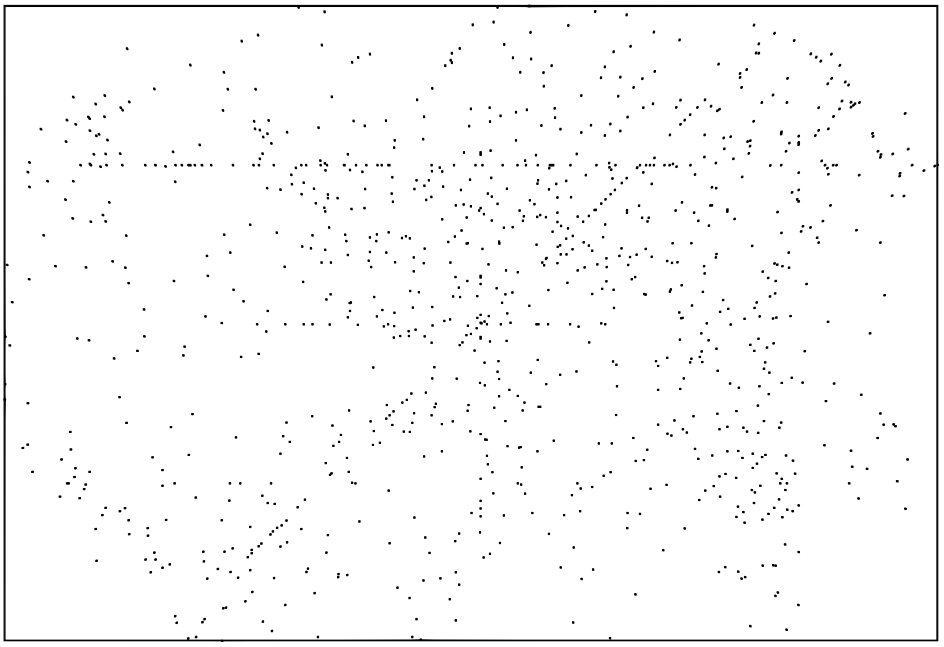
Figure 4.3: Coverage for optimal global 1-bit quantization (carpark NC dataset).



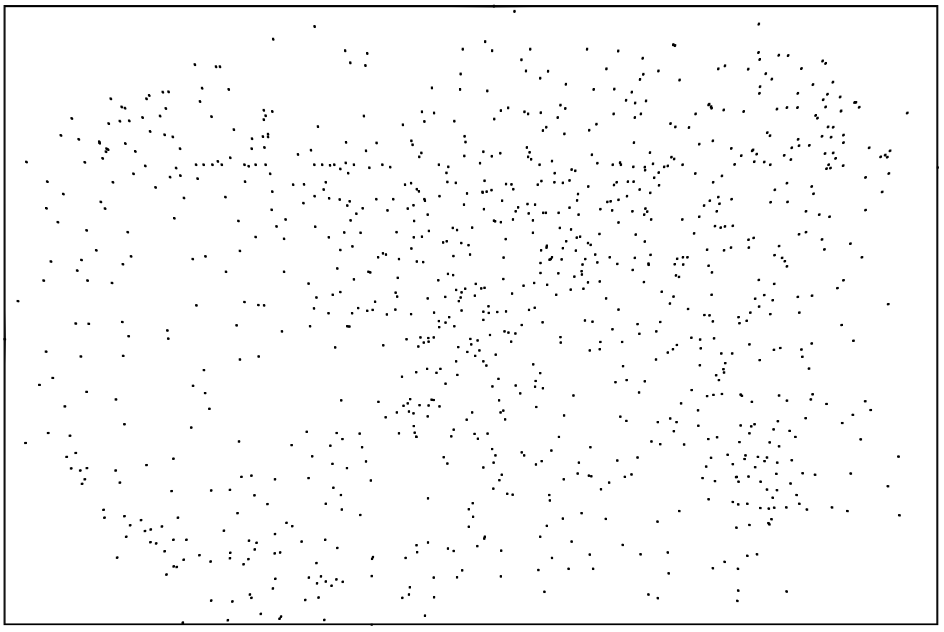
(a) RADAR



(b) LANDMARC

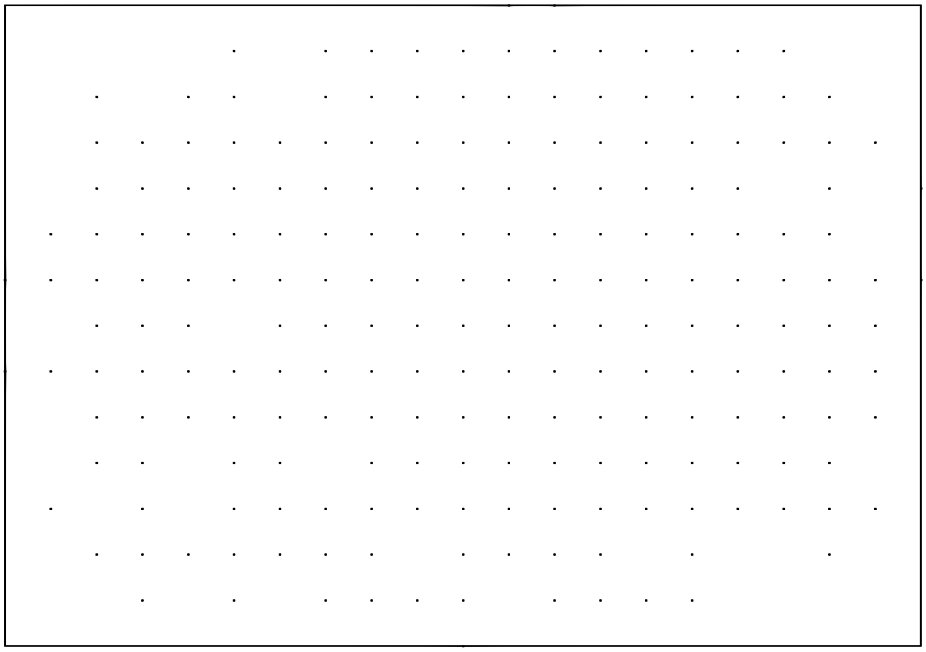


(c) LEMON

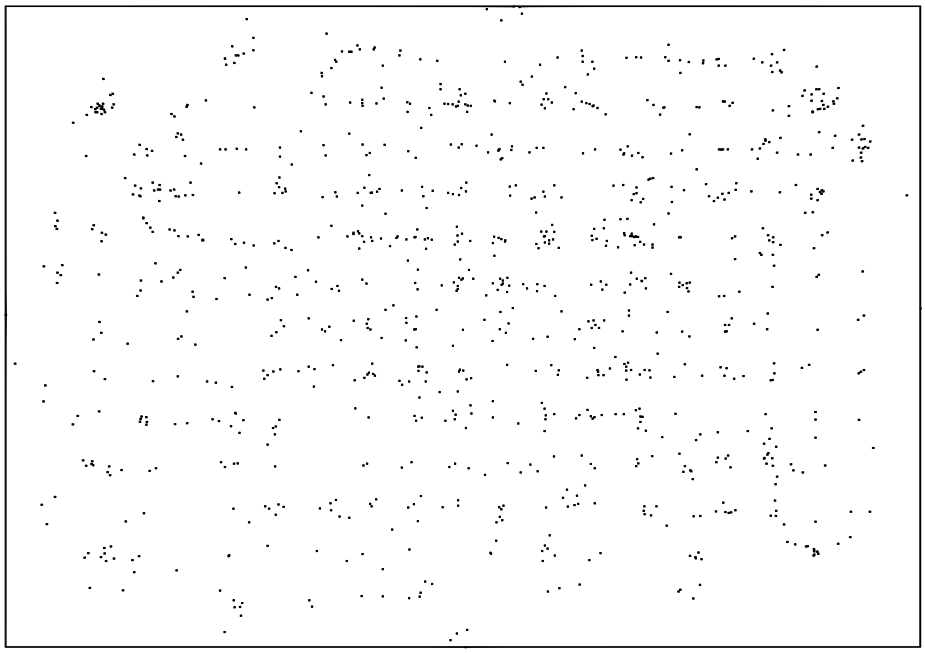


(d) LEMON ($k = 5$)

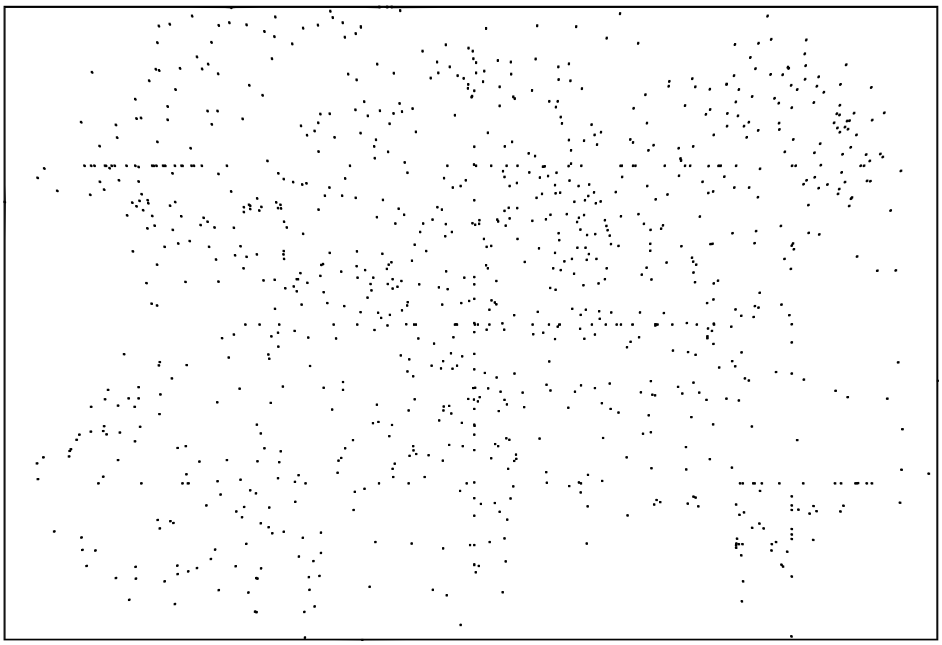
Figure 4.4: Coverage for optimal global 1-bit quantization (carpark CL dataset).



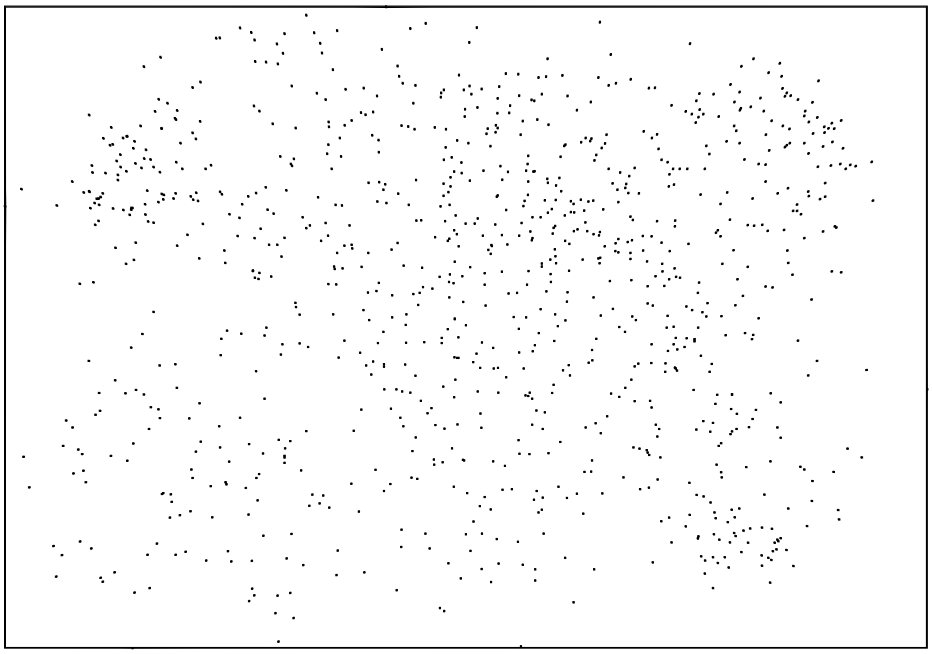
(a) RADAR



(b) LANDMARC

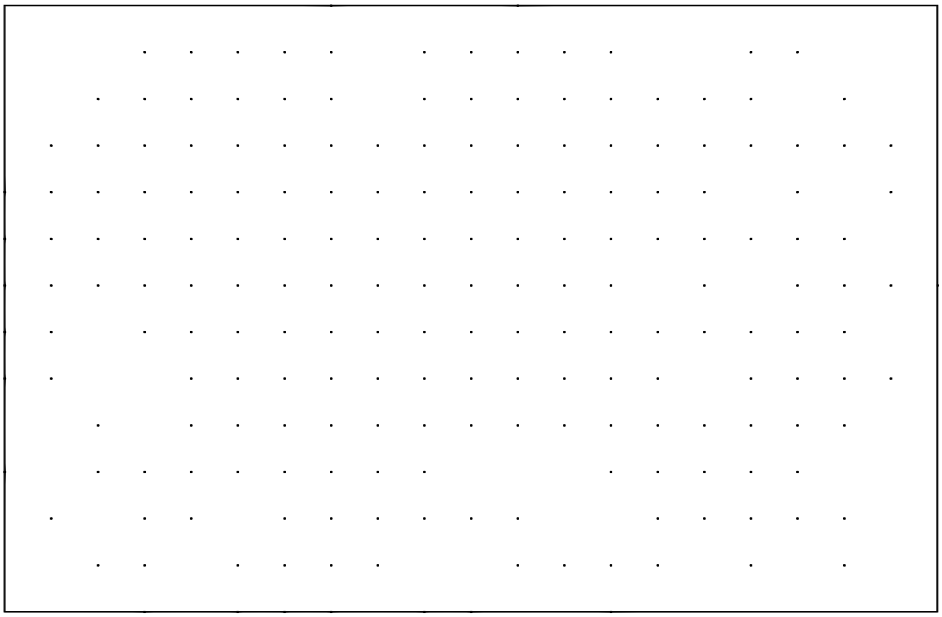


(c) LEMON

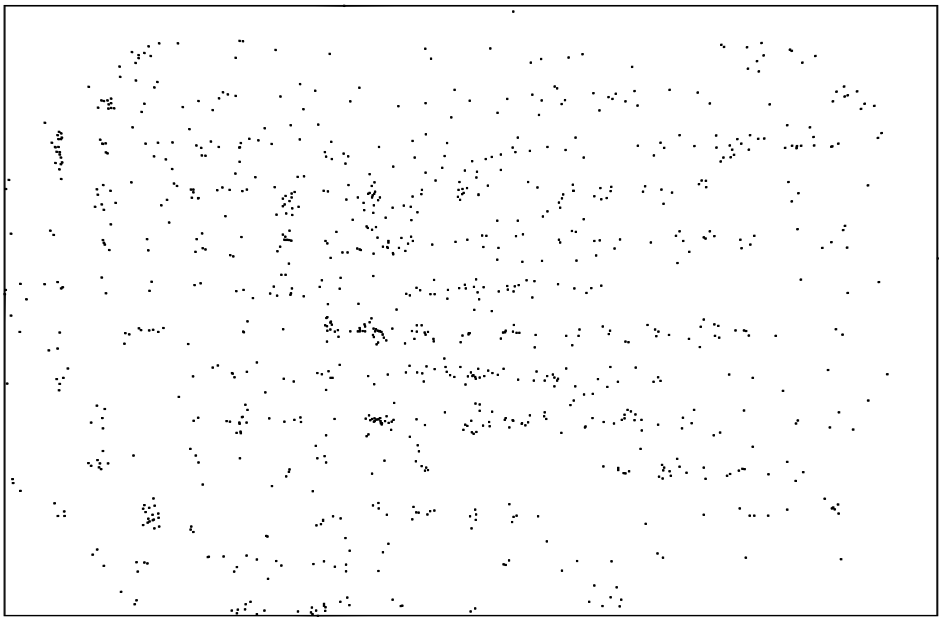


(d) LEMON ($k = 5$)

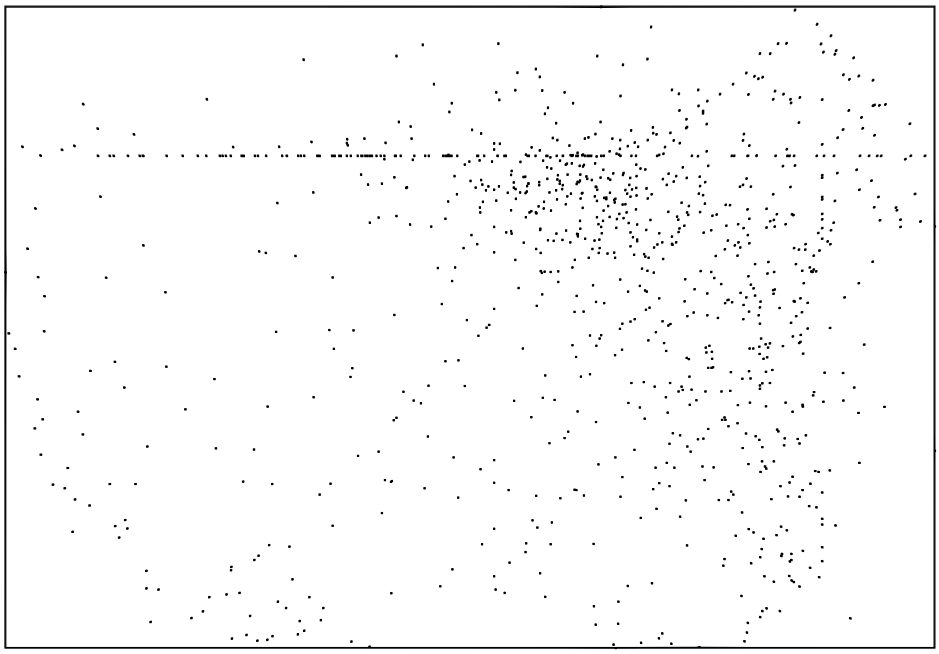
Figure 4.5: Coverage for optimal local 1-bit quantization (carpark NC dataset).



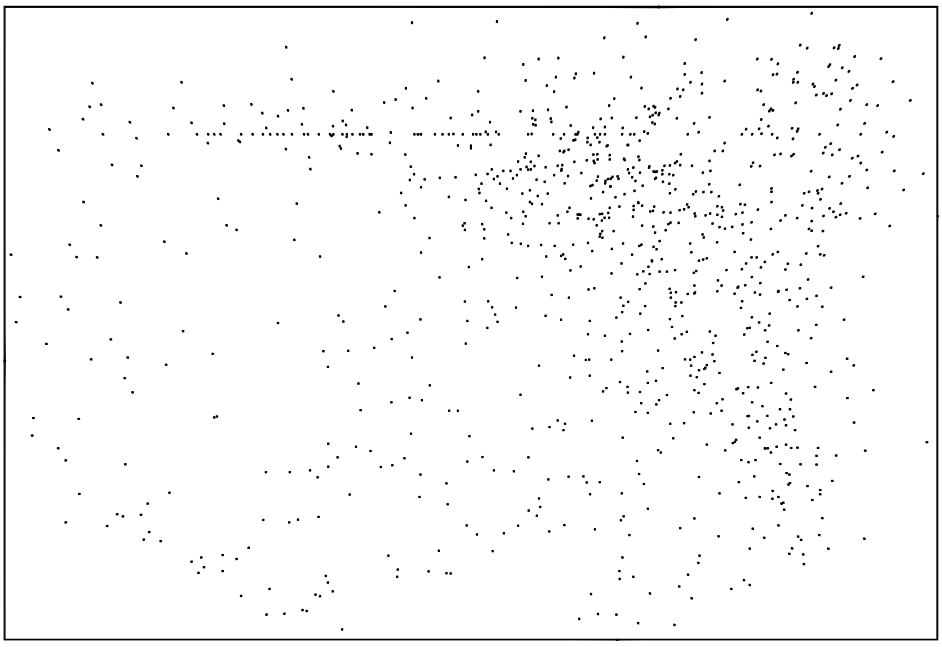
(a) RADAR



(b) LANDMARC



(c) LEMON



(d) LEMON ($k = 5$)

Figure 4.6: Coverage for optimal local 1-bit quantization (carpark CL dataset).

Chapter 5

Multiple Bit Quantization

From the results in Chapter 4, we can conclude that 1-bit quantization offers a possibly acceptably tradeoff for the purpose of localization, especially if the quantization is performed in a per-peg ("local") basis. There are shortcomings evident when the quantized data derived from training in a particular environment are used in a different/modified environment. By expanding the number of bits for the quantization, we wish to determine if there is a further improvement insofar the localization accuracy is concerned, and what could be deemed as an "adequate" number of bits to be, hopefully, as good in terms of localization performance as without quantization. Furthermore, the combination of more quantization bits might mitigate the shortcomings when the testing environment is different from the training environment.

5.1 Methodology

While an exhaustive search algorithm is sufficient for 1-bit global quantization, the search space quickly explodes when more bits are used. Assume there are N possible threshold values for any particular peg. Then, while the search space involves just N values for 1-bit global quantization and nN values with n pegs, that grows to $N(N-1)/2$ values for 2-bit global quantization, and $n \cdot N(N-1)/2$ values for 2-bit local quantization. For b bits, global quantization involves a search space of $\prod_{i=0}^{b-1}(N-i) \cdot 1/b!$, with local quantization

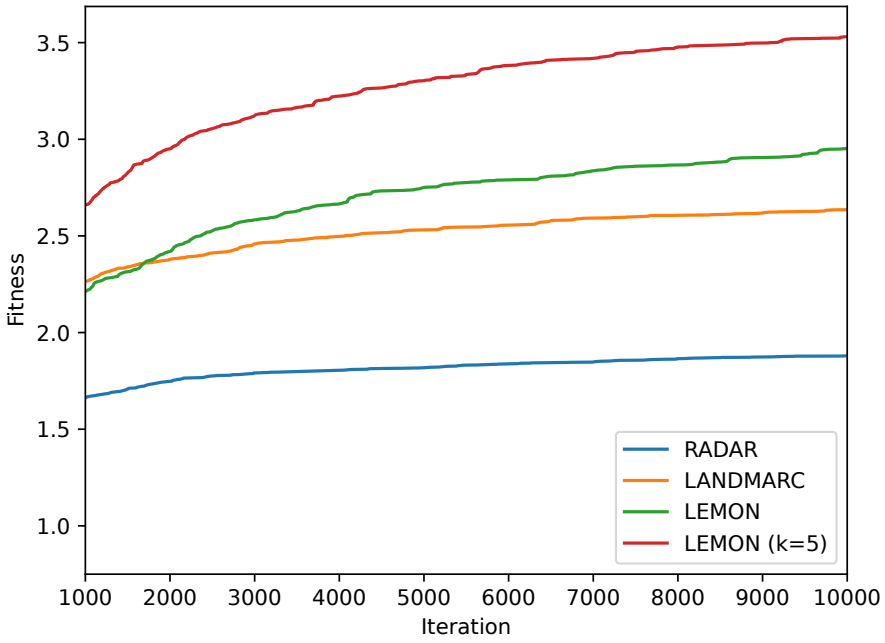


Figure 5.1: F vs. Tabu iteration (carpark NC dataset, local 2-bit quantization).

requiring a search space of $n/b! \cdot \prod_{i=0}^{b-1}(N-i)$. We expect that with a larger number of bits, higher localization accuracy can be achieved, but the tradeoff here is having to search through an increasing number of threshold combinations. At some point, exhaustive search will no longer find a solution in an acceptable timeframe, so more sophisticated search techniques are required.

As described in Chapter 2, we use Tabu search as expressed in the pseudocode of Algorithm 1 and 2 with MAX_NEIGHBOURS set to 2000. In all other respects the methodology followed is the same as in Chapter 4, with obvious generalization to have more quantization bits, and corresponding representative values. In every case for 2-bit quantization we use a short-term Tabu list of 500 entries, with a long_term_threshold value of 250, and 10000 runs at minimum. In every case for 3-bit quantization we use a short-term Tabu list of 500 entries, with a long_term_threshold value of 250, and 20000 runs at minimum. In every case of 4-bit quantization, we use a Tabu list of 1000 solutions, we create 5000 neighbours each iteration, and we let the

search run for 40000 iterations at minimum.

Evidence to the effectiveness of Tabu search can be seen in Figure 5.1 showing the increase in fitness with increasing Tabu search iterations. Note that in Figure 5.1 the first 1000 iterations of Tabu search have been removed to better compare the improvement of later iterations. Each line represents the mean fitness of 1000 runs of Tabu search.

5.2 Results

5.2.1 Carpark Dataset

For 2-, 3-, and 4-bit quantization performance is shown in Tables 5.1-5.6. A first observation is that, in general, for all three k-NN localization schemes, localization performance improves with the number of bits. The improvement possible is reduced with the more bits used, i.e., a case of diminishing returns. Also, while this improvement is evident for global quantization across all combination of training and testing combinations, the improvement for local quantization is conditioned on whether testing was performed in the same environment with training. In short, our first main observation is that local quantization strategies may be over-specializing the quantization boundaries to the point that the "overfit" for the training data set, making them problematic when subjected to a different testing set.

While a worse performance for testing different than when one trains is evident in global quantization as well, it nevertheless improves with the number of quantization bits. The same cannot be said for local quantization. For example for LEMON with $k = 5$ (a generally best performing scheme) the CR/CM (train/test) pair for local quantization moves from 1.70 (1.19) for 2-bit to 1.89 (1.41) for 4-bit. The same is not happening for global quantization where we move from 2.16 (1.36) for 2-bit to 1.78 (1.31). Also while LEMON ($k = 5$) manages to produce the best result of 0.29 (0.22) for CL/CL and 4-bits, for testing under training it does not have any notable advantage over the other schemes, where even RADAR is sometimes a competitive option!

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	0.94 (0.52)	0.91 (0.46)	0.94 (0.47)	0.94 (0.44)
	CL	1.44 (0.58)	1.29 (0.58)	1.17 (0.67)	1.20 (0.63)
	CM	1.39 (0.96)	1.33 (0.90)	1.76 (1.14)	1.69 (1.11)
	CR	1.23 (0.60)	1.14 (0.61)	1.25 (0.64)	1.20 (0.59)
CL	NC	1.31 (0.62)	1.14 (0.47)	1.46 (0.99)	1.13 (0.58)
	CL	0.97 (0.52)	0.95 (0.44)	1.05 (0.57)	1.01 (0.53)
	CM	1.55 (0.97)	1.55 (1.07)	1.67 (1.30)	1.53 (1.20)
	CR	1.57 (0.87)	1.49 (0.89)	1.54 (1.16)	1.56 (1.03)
CM	NC	1.38 (0.87)	1.37 (0.83)	1.43 (0.70)	1.38 (0.79)
	CL	1.40 (0.89)	1.39 (0.85)	1.43 (0.85)	1.38 (0.83)
	CM	1.31 (0.65)	1.29 (0.63)	1.17 (0.64)	1.23 (0.65)
	CR	1.39 (0.76)	1.39 (0.73)	1.58 (0.70)	1.47 (0.68)
CR	NC	1.46 (0.82)	1.39 (0.83)	1.49 (0.84)	1.38 (0.79)
	CL	1.66 (0.91)	1.60 (0.92)	1.67 (0.86)	1.61 (1.01)
	CM	2.08 (1.20)	2.05 (1.24)	2.01 (1.46)	2.16 (1.36)
	CR	1.14 (0.59)	1.08 (0.55)	1.00 (0.44)	1.05 (0.45)

Table 5.1: Localization performance (carpark dataset, global 2-bit quantization).

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	0.60 (0.42)	0.51 (0.28)	0.46 (0.39)	0.42 (0.34)
	CL	1.32 (0.49)	1.44 (0.78)	1.18 (0.95)	1.27 (0.81)
	CM	1.58 (0.93)	1.61 (1.00)	1.77 (1.17)	1.47 (1.13)
	CR	1.26 (0.75)	1.42 (0.87)	1.43 (0.82)	1.29 (0.62)
CL	NC	1.39 (0.63)	1.25 (0.75)	1.33 (0.78)	1.22 (0.70)
	CL	0.64 (0.38)	1.57 (0.27)	0.43 (0.22)	1.46 (0.23)
	CM	1.65 (1.09)	1.83 (1.23)	1.73 (1.07)	1.58 (1.07)
	CR	1.59 (0.88)	1.62 (0.75)	1.79 (1.20)	1.44 (0.77)
CM	NC	1.58 (0.73)	1.17 (0.66)	1.46 (0.98)	1.28 (0.66)
	CL	1.43 (0.54)	1.40 (0.82)	1.59 (0.81)	1.53 (0.66)
	CM	0.82 (0.54)	0.79 (0.48)	0.74 (0.43)	0.69 (0.43)
	CR	1.61 (0.95)	1.29 (0.64)	1.33 (0.96)	1.33 (0.62)
CR	NC	1.50 (0.73)	1.47 (0.86)	1.81 (1.35)	1.62 (0.94)
	CL	1.59 (0.96)	1.62 (0.83)	1.91 (1.26)	1.76 (0.84)
	CM	1.64 (1.09)	1.87 (1.34)	1.89 (1.39)	1.70 (1.19)
	CR	0.59 (0.40)	0.54 (0.34)	0.44 (0.25)	0.46 (0.27)

Table 5.2: Localization performance (carpark dataset, local 2-bit quantization).

train	**test**	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	0.80 (0.44)	0.76 (0.41)	0.80 (0.40)	0.76 (0.39)
	CL	1.12 (0.49)	1.05 (0.52)	1.08 (0.58)	1.05 (0.57)
	CM	1.40 (0.98)	1.31 (0.93)	1.61 (1.15)	1.54 (1.12)
	CR	1.17 (0.48)	1.06 (0.52)	1.17 (0.63)	1.09 (0.58)
CL	NC	1.04 (0.56)	1.01 (0.55)	0.98 (0.71)	0.96 (0.63)
	CL	0.86 (0.40)	0.76 (0.39)	0.79 (0.34)	0.77 (0.34)
	CM	1.47 (0.96)	1.47 (0.99)	1.37 (1.05)	1.43 (1.06)
	CR	1.19 (0.61)	1.28 (0.64)	1.35 (0.80)	1.31 (0.78)
CM	NC	1.12 (0.60)	1.09 (0.61)	1.15 (0.74)	1.03 (0.65)
	CL	1.21 (0.67)	1.21 (0.75)	1.40 (0.82)	1.24 (0.77)
	CM	1.14 (0.62)	1.07 (0.61)	1.06 (0.54)	1.02 (0.57)
	CR	1.24 (0.55)	1.14 (0.54)	1.21 (0.64)	1.19 (0.57)
CR	NC	1.33 (0.82)	1.26 (0.86)	1.27 (0.77)	1.19 (0.79)
	CL	1.29 (0.74)	1.29 (0.77)	1.44 (0.95)	1.38 (0.91)
	CM	1.70 (1.10)	1.63 (1.11)	1.89 (1.33)	1.80 (1.26)
	CR	0.91 (0.36)	0.90 (0.36)	0.86 (0.39)	0.87 (0.40)

Table 5.3: Localization performance (carpark dataset, global 3-bit quantization).

train	**test**	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	0.56 (0.42)	0.46 (0.31)	0.44 (0.39)	0.41 (0.31)
	CL	1.25 (0.68)	0.99 (0.55)	1.21 (0.69)	1.03 (0.68)
	CM	1.64 (1.02)	1.42 (0.81)	1.58 (1.20)	1.44 (1.07)
	CR	1.15 (0.49)	1.12 (0.60)	1.32 (0.70)	1.20 (0.59)
CL	NC	1.20 (0.55)	1.17 (0.64)	1.26 (0.68)	1.14 (0.75)
	CL	0.59 (0.39)	0.42 (0.27)	0.32 (0.26)	0.31 (0.23)
	CM	1.53 (1.08)	1.60 (1.10)	1.76 (1.21)	1.56 (1.05)
	CR	1.49 (0.76)	1.36 (0.80)	1.57 (0.99)	1.37 (0.55)
CM	NC	1.26 (0.70)	1.28 (0.61)	1.25 (0.81)	1.23 (0.73)
	CL	1.36 (0.72)	1.41 (0.64)	1.51 (0.77)	1.44 (0.70)
	CM	0.86 (0.53)	0.71 (0.46)	0.60 (0.44)	0.61 (0.44)
	CR	1.27 (0.63)	1.23 (0.64)	1.30 (0.80)	1.21 (0.67)
CR	NC	1.65 (0.94)	1.15 (0.71)	1.23 (0.82)	1.40 (0.87)
	CL	1.42 (0.85)	1.47 (0.85)	1.60 (1.11)	1.64 (1.10)
	CM	1.98 (1.23)	1.75 (1.14)	1.80 (1.60)	1.99 (1.37)
	CR	0.62 (0.38)	0.48 (0.33)	0.34 (0.26)	0.38 (0.25)

Table 5.4: Localization performance (carpark dataset, local 3-bit quantization).

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	0.78 (0.43)	0.73 (0.42)	0.79 (0.38)	0.74 (0.40)
	CL	1.03 (0.55)	1.02 (0.50)	1.06 (0.75)	1.00 (0.50)
	CM	1.38 (0.84)	1.25 (0.85)	1.52 (1.02)	1.47 (1.08)
	CR	1.23 (0.54)	1.04 (0.57)	1.18 (0.59)	1.03 (0.55)
CL	NC	1.07 (0.56)	0.83 (0.59)	0.99 (0.69)	0.98 (0.67)
	CL	0.81 (0.42)	0.72 (0.42)	0.77 (0.34)	0.75 (0.36)
	CM	1.50 (1.00)	1.40 (1.04)	1.37 (1.04)	1.40 (1.04)
	CR	1.28 (0.63)	1.22 (0.59)	1.37 (0.80)	1.29 (0.76)
CM	NC	1.15 (0.62)	1.12 (0.61)	1.17 (0.72)	1.05 (0.65)
	CL	1.24 (0.69)	1.28 (0.73)	1.38 (0.80)	1.29 (0.77)
	CM	1.14 (0.65)	1.06 (0.65)	1.02 (0.55)	1.00 (0.60)
	CR	1.23 (0.55)	1.20 (0.51)	1.20 (0.61)	1.22 (0.55)
CR	NC	1.36 (0.88)	1.28 (0.92)	1.28 (0.84)	1.15 (0.74)
	CL	1.16 (0.67)	1.11 (0.71)	1.43 (0.95)	1.26 (0.80)
	CM	1.78 (1.11)	1.68 (1.15)	1.86 (1.30)	1.78 (1.31)
	CR	0.87 (0.37)	0.83 (0.35)	0.83 (0.39)	0.84 (0.42)

Table 5.5: Localization performance (carpark dataset, global 4-bit quantization).

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	0.63 (0.41)	0.45 (0.34)	0.38 (0.43)	0.40 (0.33)
	CL	1.08 (0.52)	0.97 (0.58)	1.35 (0.87)	1.00 (0.53)
	CM	1.71 (0.80)	1.45 (0.94)	1.59 (1.25)	1.40 (1.01)
	CR	1.12 (0.57)	1.22 (0.61)	1.26 (0.64)	1.11 (0.57)
CL	NC	1.20 (0.52)	0.99 (0.68)	1.04 (0.67)	1.09 (0.69)
	CL	0.70 (0.48)	0.41 (0.27)	0.38 (0.22)	0.29 (0.22)
	CM	1.54 (1.02)	1.54 (1.09)	1.53 (1.13)	1.52 (1.14)
	CR	1.31 (0.63)	1.37 (0.81)	1.31 (0.67)	1.21 (0.53)
CM	NC	1.14 (0.69)	1.20 (0.82)	1.17 (0.79)	1.12 (0.67)
	CL	1.34 (0.64)	1.37 (0.72)	1.43 (0.73)	1.35 (0.64)
	CM	0.98 (0.58)	0.80 (0.54)	0.54 (0.43)	0.62 (0.47)
	CR	1.27 (0.82)	1.34 (0.73)	1.26 (0.66)	1.09 (0.55)
CR	NC	1.32 (0.89)	1.16 (0.91)	1.18 (0.79)	1.13 (0.74)
	CL	1.38 (0.76)	1.42 (0.96)	1.47 (1.02)	1.56 (1.05)
	CM	1.80 (1.20)	1.83 (1.26)	1.64 (1.25)	1.89 (1.41)
	CR	0.69 (0.38)	0.48 (0.36)	0.40 (0.31)	0.39 (0.28)

Table 5.6: Localization performance (carpark dataset, local 4-bit quantization).

Another significant result is that when we compare the results here with the case of no quantization (Table 4.2), we note that with increasing number of bits for quantization, multi-bit quantization can actually outperform localization without quantization! Again this observation is conditional on whether testing and training is on the same environment – yet when they are different there seems to be notable advantage to having full resolution measurements either. The improvement over full resolution RSSI measurements may be partly explained in the "binning" done by quantization, and the corresponding representative values of the bins, that remove minor fluctuations in RSSI values, acting as a "filter" to noise present in the full resolution measurements.

A less apparent trend is that as we move to 3-bit quantization, there is a modest improvement compared to 2-bits, especially with the weighted k-NN algorithms LANDMARC and LEMON. As with local 2-bit quantization, LANDMARC and LEMON also better take advantage of the higher number of bits compared to RADAR. Moving to 4-bits does not notably enhance this difference, suggesting again a point of diminishing returns.

Given the above observations, one could start questioning whether it is worth to derive specific quantization boundaries that fit a particular dataset. To this end, we will see next how the fixed quantization boundaries fare.

5.2.2 Fixed Quantization

As in the 1-bit scenario, we also test localization accuracy using the fixed quantization boundaries described in equations 2.6 – 2.9. For the sake of completeness we provide the fixed quantization performance for all four fixed quantizers (linear, square-root, cubic-root, log) and for 2-, 3-, and 4- bit quantization. The results are shown in Tables 5.7–5.18. As a final observation, we again find that the linear quantizer often results in good performance.

A general observation is that, even with increasing bits, the localization performance may not demonstrate a trends of improving, and indeed it can

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.13 (0.52)	1.05 (0.51)	1.03 (0.62)	1.04 (0.51)
	CL	1.27 (0.64)	1.21 (0.63)	1.34 (0.92)	1.25 (0.78)
	CM	1.69 (1.08)	1.69 (1.11)	1.78 (1.34)	1.74 (1.25)
	CR	1.55 (0.66)	1.50 (0.61)	1.55 (0.66)	1.50 (0.58)
CL	NC	1.18 (0.52)	1.02 (0.59)	1.15 (0.75)	1.07 (0.69)
	CL	1.34 (0.80)	1.28 (0.88)	1.56 (1.10)	1.43 (1.00)
	CM	1.81 (0.97)	1.75 (0.99)	1.69 (1.08)	1.71 (1.04)
	CR	1.58 (0.70)	1.54 (0.78)	1.76 (1.23)	1.60 (1.06)
CM	NC	1.27 (0.58)	1.16 (0.66)	1.31 (0.81)	1.17 (0.73)
	CL	1.59 (1.00)	1.54 (1.03)	1.62 (1.09)	1.52 (1.06)
	CM	1.69 (1.02)	1.62 (1.07)	1.62 (1.22)	1.61 (1.20)
	CR	1.49 (0.82)	1.46 (0.81)	1.59 (0.84)	1.49 (0.77)
CR	NC	1.39 (0.74)	1.32 (0.74)	1.35 (0.80)	1.29 (0.79)
	CL	1.33 (0.81)	1.30 (0.77)	1.43 (0.84)	1.32 (0.81)
	CM	1.86 (1.10)	1.81 (1.15)	2.09 (1.73)	1.95 (1.52)
	CR	1.44 (0.67)	1.39 (1.60)	1.32 (0.62)	1.31 (0.54)

Table 5.7: Localization performance (carpark dataset, linear 2-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.25 (0.71)	1.21 (0.74)	1.40 (0.88)	1.29 (0.83)
	CL	1.23 (0.70)	1.18 (0.67)	1.30 (0.84)	1.20 (0.77)
	CM	1.73 (1.10)	1.69 (1.11)	1.94 (1.39)	1.76 (1.21)
	CR	1.47 (0.89)	1.41 (0.89)	1.52 (0.95)	1.41 (0.87)
CL	NC	1.50 (0.70)	1.40 (0.75)	1.43 (0.91)	1.40 (0.88)
	CL	1.35 (0.99)	1.31 (0.97)	1.36 (0.92)	1.28 (0.93)
	CM	1.79 (1.14)	1.74 (1.14)	1.76 (1.17)	1.72 (1.14)
	CR	1.52 (0.89)	1.47 (0.86)	1.63 (1.08)	1.48 (0.96)
CM	NC	1.51 (0.82)	1.44 (0.75)	1.31 (0.72)	1.35 (0.70)
	CL	1.34 (0.65)	1.29 (0.61)	1.45 (0.74)	1.35 (0.64)
	CM	1.70 (0.94)	1.59 (0.92)	1.56 (0.94)	1.53 (0.96)
	CR	1.28 (0.70)	1.28 (0.67)	1.56 (0.76)	1.46 (0.67)
CR	NC	1.38 (0.81)	1.29 (0.83)	1.35 (0.90)	1.26 (0.88)
	CL	1.51 (0.89)	1.40 (0.81)	1.64 (1.05)	1.39 (0.83)
	CM	1.99 (1.17)	1.92 (1.17)	1.78 (1.06)	1.80 (1.09)
	CR	1.57 (0.78)	1.47 (0.78)	1.35 (0.91)	1.32 (0.85)

Table 5.8: Localization performance (carpark dataset, $\sqrt[2]{}$ 2-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.54 (0.99)	1.48 (0.96)	1.53 (1.03)	1.48 (0.99)
	CL	1.32 (0.72)	1.26 (0.67)	1.19 (0.67)	1.21 (0.63)
	CM	1.71 (0.97)	1.65 (1.00)	1.74 (1.16)	1.61 (1.09)
	CR	1.29 (0.63)	1.24 (0.58)	1.40 (0.68)	1.28 (0.61)
CL	NC	1.50 (0.96)	1.36 (1.02)	1.35 (0.99)	1.30 (1.03)
	CL	1.23 (0.69)	1.20 (0.72)	1.36 (0.89)	1.30 (0.82)
	CM	1.71 (0.87)	1.65 (0.90)	1.69 (1.13)	1.56 (1.02)
	CR	1.42 (0.56)	1.36 (0.54)	1.53 (0.81)	1.42 (0.72)
CM	NC	1.38 (0.99)	1.26 (0.96)	0.24 (0.86)	1.18 (0.91)
	CL	1.61 (0.64)	1.54 (0.65)	1.58 (0.79)	1.56 (0.75)
	CM	1.53 (0.73)	1.46 (0.75)	1.48 (0.92)	1.50 (0.89)
	CR	1.26 (0.58)	1.22 (0.53)	0.40 (0.76)	0.34 (0.62)
CR	NC	1.41 (0.95)	0.36 (0.97)	1.44 (1.12)	1.38 (1.08)
	CL	1.58 (0.81)	1.54 (0.79)	1.60 (1.13)	1.54 (0.98)
	CM	1.70 (1.02)	1.64 (1.03)	1.69 (1.04)	1.65 (1.00)
	CR	1.61 (0.73)	1.50 (0.72)	1.40 (0.87)	1.42 (0.76)

Table 5.9: Localization performance (carpark dataset, $\sqrt[3]{}$ 2-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.52 (0.87)	1.45 (0.89)	1.51 (0.96)	1.43 (0.94)
	CL	1.48 (0.72)	1.44 (0.73)	1.36 (0.78)	1.37 (0.72)
	CM	1.90 (0.19)	1.88 (1.22)	1.91 (1.38)	1.87 (1.31)
	CR	1.60 (0.86)	1.49 (0.90)	1.54 (1.20)	1.42 (0.92)
CL	NC	1.58 (0.90)	1.48 (0.97)	1.54 (1.11)	1.48 (1.05)
	CL	1.46 (0.97)	0.39 (0.94)	1.51 (0.91)	1.41 (0.89)
	CM	1.77 (1.07)	1.74 (1.08)	1.78 (1.25)	1.71 (1.17)
	CR	1.68 (0.89)	1.62 (0.88)	1.60 (1.03)	1.58 (0.98)
CM	NC	1.67 (1.01)	1.60 (0.99)	1.50 (0.85)	1.54 (0.87)
	CL	1.56 (0.58)	1.53 (0.58)	1.60 (0.73)	1.57 (0.68)
	CM	1.72 (0.81)	1.62 (0.77)	1.52 (0.88)	1.52 (0.82)
	CR	1.49 (0.71)	1.45 (0.72)	1.62 (0.84)	1.52 (0.73)
CR	NC	1.64 (0.83)	1.57 (0.88)	1.67 (0.95)	1.58 (0.94)
	CL	1.49 (0.89)	1.45 (0.86)	1.60 (1.01)	1.48 (0.96)
	CM	2.09 (1.27)	2.07 (1.26)	2.09 (1.34)	2.10 (1.27)
	CR	1.54 (0.76)	1.47 (0.70)	1.42 (0.83)	1.40 (0.76)

Table 5.10: Localization performance (carpark dataset, log 2-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.14 (0.64)	1.09 (0.60)	1.13 (0.61)	1.12 (0.56)
	CL	1.08 (0.57)	1.00 (0.44)	1.10 (0.62)	1.04 (0.52)
	CM	1.52 (0.81)	1.52 (0.85)	1.76 (1.21)	1.61 (1.05)
	CR	1.33 (0.59)	1.26 (0.55)	1.34 (0.58)	1.28 (0.58)
CL	NC	1.15 (0.59)	0.99 (0.64)	1.03 (0.67)	0.98 (0.62)
	CL	1.09 (0.65)	0.99 (0.63)	1.11 (0.64)	0.97 (0.63)
	CM	1.51 (0.93)	1.41 (0.97)	1.49 (1.04)	1.43 (0.98)
	CR	1.28 (0.59)	1.20 (0.66)	1.37 (0.91)	1.28 (0.79)
CM	NC	1.19 (0.74)	1.09 (0.71)	1.13 (0.76)	1.09 (0.77)
	CL	1.16 (0.67)	1.11 (0.65)	1.42 (0.77)	1.26 (0.69)
	CM	1.50 (0.82)	1.38 (0.84)	1.28 (0.83)	1.31 (0.82)
	CR	1.26 (0.73)	1.22 (0.70)	1.39 (0.83)	1.24 (0.70)
CR	NC	1.27 (0.82)	1.19 (0.75)	1.19 (0.60)	1.14 (0.62)
	CL	1.20 (0.76)	1.21 (0.80)	1.49 (1.09)	1.38 (0.98)
	CM	1.83 (0.95)	1.73 (0.99)	1.70 (1.10)	1.63 (1.06)
	CR	1.32 (0.66)	1.21 (0.62)	1.07 (0.55)	1.09 (0.57)

Table 5.11: Localization performance (carpark dataset, linear 3-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.21 (0.69)	1.13 (0.63)	1.34 (0.67)	1.15 (0.65)
	CL	1.28 (0.61)	1.29 (0.77)	1.26 (1.07)	1.24 (0.90)
	CM	1.44 (0.94)	1.40 (0.96)	1.53 (1.08)	1.42 (1.03)
	CR	1.24 (0.57)	1.19 (0.58)	1.41 (0.78)	1.30 (0.66)
CL	NC	1.17 (0.46)	1.04 (0.49)	1.07 (0.63)	1.02 (0.59)
	CL	1.23 (0.79)	1.15 (0.82)	1.31 (0.84)	1.21 (0.78)
	CM	1.58 (1.02)	1.48 (1.05)	1.53 (1.24)	1.44 (1.09)
	CR	1.55 (0.74)	1.49 (0.83)	1.64 (1.21)	1.58 (1.07)
CM	NC	1.37 (0.70)	1.31 (0.67)	1.37 (0.71)	1.30 (0.70)
	CL	1.46 (0.97)	1.36 (0.98)	1.50 (0.97)	1.41 (1.00)
	CM	1.52 (0.88)	1.42 (0.87)	1.31 (0.83)	1.34 (0.84)
	CR	1.44 (0.87)	1.42 (0.78)	1.46 (0.81)	1.41 (0.76)
CR	NC	1.62 (0.92)	1.53 (0.95)	1.46 (1.08)	1.45 (1.04)
	CL	1.67 (0.93)	1.54 (1.01)	1.63 (1.13)	1.46 (1.05)
	CM	1.82 (1.09)	1.71 (1.10)	1.67 (1.22)	1.66 (1.15)
	CR	1.17 (0.60)	1.09 (0.58)	1.17 (0.68)	1.09 (0.59)

Table 5.12: Localization performance (carpark dataset, $\sqrt[2]{}$ 3-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.15 (0.39)	1.08 (0.49)	1.13 (0.74)	1.11 (0.69)
	CL	1.07 (0.56)	1.05 (0.53)	1.13 (0.70)	1.03 (0.59)
	CM	1.46 (0.85)	1.45 (0.89)	1.54 (1.09)	1.53 (1.03)
	CR	1.41 (0.87)	1.31 (0.83)	1.28 (0.90)	1.29 (0.87)
CL	NC	1.21 (0.72)	1.02 (0.74)	1.18 (0.79)	1.00 (0.78)
	CL	1.15 (0.59)	1.04 (0.63)	1.13 (0.74)	1.06 (0.69)
	CM	1.42 (0.95)	1.35 (0.98)	1.47 (1.08)	1.39 (1.06)
	CR	1.42 (0.75)	1.31 (0.79)	1.48 (1.00)	1.38 (0.98)
CM	NC	1.39 (0.76)	1.29 (0.73)	1.32 (0.92)	1.32 (0.83)
	CL	1.41 (0.74)	1.31 (0.67)	1.35 (0.66)	1.26 (0.69)
	CM	1.61 (0.74)	1.54 (1.72)	1.52 (0.90)	1.50 (0.82)
	CR	1.32 (0.84)	1.29 (0.77)	1.31 (0.76)	1.31 (0.74)
CR	NC	1.24 (0.91)	1.16 (0.88)	1.22 (0.73)	1.08 (0.82)
	CL	1.31 (0.76)	1.28 (0.79)	1.50 (1.05)	1.38 (0.99)
	CM	1.60 (1.08)	1.57 (1.11)	1.74 (1.38)	1.62 (1.24)
	CR	1.26 (0.54)	1.17 (0.52)	1.17 (0.54)	1.12 (0.52)

Table 5.13: Localization performance (carpark dataset, $\sqrt[3]{}$ 3-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.24 (0.49)	1.17 (0.59)	1.26 (0.72)	1.20 (0.65)
	CL	1.17 (0.57)	1.02 (0.48)	1.05 (0.59)	0.97 (0.55)
	CM	1.44 (0.86)	1.43 (0.91)	1.61 (1.16)	1.54 (1.08)
	CR	1.28 (0.76)	1.17 (0.74)	1.21 (0.73)	1.11 (0.71)
CL	NC	1.20 (0.67)	1.05 (0.68)	1.24 (0.80)	1.11 (0.77)
	CL	1.09 (0.48)	1.02 (0.48)	1.19 (0.73)	1.05 (0.57)
	CM	1.48 (0.95)	1.41 (0.99)	1.63 (1.20)	1.43 (1.08)
	CR	1.51 (1.78)	1.38 (0.76)	1.48 (0.85)	1.36 (0.73)
CM	NC	1.33 (0.73)	1.27 (0.68)	1.25 (0.85)	1.25 (0.76)
	CL	1.40 (0.58)	1.36 (0.60)	1.52 (0.68)	1.37 (0.60)
	CM	1.62 (0.73)	1.56 (0.72)	1.51 (0.81)	1.51 (0.73)
	CR	0.35 (0.77)	1.26 (0.73)	1.32 (0.74)	1.25 (0.70)
CR	NC	1.25 (0.87)	1.18 (0.87)	1.21 (0.78)	1.15 (0.81)
	CL	1.43 (0.85)	1.39 (0.86)	1.58 (1.08)	1.46 (1.00)
	CM	1.72 (1.20)	1.64 (1.20)	1.68 (1.32)	1.59 (1.29)
	CR	1.33 (0.65)	1.25 (0.60)	1.17 (0.57)	1.18 (0.54)

Table 5.14: Localization performance (carpark dataset, log 3-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.20 (0.49)	1.14 (0.53)	1.17 (0.65)	1.12 (0.58)
	CL	1.04 (0.60)	0.96 (0.59)	1.03 (0.63)	0.94 (0.58)
	CM	1.54 (0.89)	1.53 (0.96)	1.72 (1.25)	1.61 (1.12)
	CR	1.37 (0.63)	1.21 (0.57)	1.24 (0.69)	1.19 (0.63)
CL	NC	1.26 (0.55)	1.19 (0.57)	1.18 (0.67)	1.14 (0.63)
	CL	1.02 (0.48)	0.93 (0.51)	1.00 (0.61)	0.95 (0.56)
	CM	1.40 (0.81)	1.39 (0.91)	1.64 (1.27)	1.50 (1.11)
	CR	1.30 (0.64)	1.18 (0.63)	1.23 (0.65)	1.15 (0.62)
CM	NC	1.22 (0.49)	1.13 (0.53)	1.12 (0.66)	1.10 (0.58)
	CL	0.98 (0.46)	0.92 (0.45)	0.98 (0.61)	0.92 (0.56)
	CM	1.46 (0.80)	1.45 (0.86)	1.65 (1.22)	1.55 (1.06)
	CR	1.30 (0.62)	1.19 (0.60)	1.26 (0.63)	1.18 (0.55)
CR	NC	1.18 (0.48)	1.11 (0.49)	1.09 (0.63)	1.08 (0.54)
	CL	1.04 (0.59)	0.97 (0.58)	0.96 (0.61)	0.94 (0.61)
	CM	1.55 (0.85)	1.51 (0.93)	1.67 (1.22)	1.54 (1.04)
	CR	1.24 (0.64)	1.16 (0.59)	1.18 (0.63)	1.16 (0.58)

Table 5.15: Localization performance (carpark dataset, linear 4-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.19 (0.53)	1.09 (0.49)	1.11 (0.69)	1.07 (0.60)
	CL	1.16 (0.59)	1.07 (0.56)	0.99 (0.64)	0.96 (0.55)
	CM	1.55 (0.94)	1.51 (0.97)	1.66 (1.18)	1.56 (1.14)
	CR	1.36 (0.95)	1.25 (0.91)	1.32 (0.84)	1.26 (0.85)
CL	NC	1.20 (0.51)	1.12 (0.59)	1.19 (0.78)	1.12 (0.69)
	CL	1.21 (0.51)	1.23 (0.53)	1.07 (0.61)	1.05 (0.53)
	CM	1.42 (0.90)	1.37 (0.94)	1.50 (1.06)	1.37 (1.01)
	CR	1.27 (0.65)	1.20 (0.66)	1.28 (0.75)	1.23 (0.70)
CM	NC	1.20 (0.48)	1.14 (0.57)	1.19 (0.72)	1.13 (0.67)
	CL	1.20 (0.51)	1.11 (0.52)	1.06 (0.56)	1.02 (0.53)
	CM	1.47 (0.85)	1.40 (0.91)	1.50 (1.14)	1.42 (1.03)
	CR	1.19 (0.50)	1.17 (0.52)	1.44 (0.80)	1.29 (0.63)
CR	NC	1.22 (0.47)	1.16 (0.54)	1.24 (0.70)	1.17 (0.63)
	CL	1.16 (0.47)	1.07 (0.50)	1.14 (0.61)	1.07 (0.56)
	CM	1.59 (0.94)	1.52 (1.00)	1.48 (1.05)	1.47 (1.04)
	CR	1.36 (0.58)	1.32 (0.55)	1.46 (0.78)	1.44 (0.70)

Table 5.16: Localization performance (carpark dataset, $\sqrt[2]{}$ 4-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.15 (0.39)	1.08 (0.49)	1.13 (0.74)	1.11 (0.69)
	CL	1.07 (0.56)	1.05 (0.53)	1.13 (0.70)	1.03 (0.59)
	CM	1.46 (0.85)	1.45 (0.89)	1.54 (1.09)	1.53 (1.03)
	CR	1.41 (0.87)	1.31 (0.83)	1.28 (0.90)	1.29 (0.87)
CL	NC	1.21 (0.72)	1.02 (0.74)	1.18 (0.79)	1.00 (0.78)
	CL	1.15 (0.59)	1.04 (0.63)	1.13 (0.74)	1.06 (0.69)
	CM	1.42 (0.95)	1.35 (0.98)	1.47 (1.08)	1.39 (1.06)
	CR	1.42 (0.75)	1.31 (0.79)	1.48 (1.00)	1.38 (0.98)
CM	NC	1.39 (0.76)	1.29 (0.73)	1.32 (0.92)	1.32 (0.83)
	CL	1.41 (0.74)	1.31 (0.67)	1.35 (0.66)	1.26 (0.69)
	CM	1.61 (0.74)	1.54 (1.72)	1.52 (0.90)	1.50 (0.82)
	CR	1.32 (0.84)	1.29 (0.77)	1.31 (0.76)	1.31 (0.74)
CR	NC	1.24 (0.91)	1.16 (0.88)	1.22 (0.73)	1.08 (0.82)
	CL	1.31 (0.76)	1.28 (0.79)	1.50 (1.05)	1.38 (0.99)
	CM	1.60 (1.08)	1.57 (1.11)	1.74 (1.38)	1.62 (1.24)
	CR	1.26 (0.54)	1.17 (0.52)	1.17 (0.54)	1.12 (0.52)

Table 5.17: Localization performance (carpark dataset, $\sqrt[3]{}$ 4-bit quantization)

train	test	$E(x)$ ($\sigma(x)$)			
		RADAR	**LANDMARC**	**LEMON**	**LEMON** (k=5)
NC	NC	1.19 (0.64)	1.10 (0.65)	1.13 (0.77)	1.07 (0.71)
	CL	1.13 (0.52)	1.05 (0.54)	1.10 (0.70)	1.02 (0.62)
	CM	1.52 (0.94)	1.45 (0.98)	1.48 (1.17)	1.43 (1.11)
	CR	1.33 (0.60)	1.25 (0.61)	1.41 (0.77)	1.29 (0.69)
CL	NC	1.34 (0.60)	1.26 (0.60)	1.29 (0.85)	1.24 (0.68)
	CL	1.00 (0.44)	0.93 (0.43)	0.98 (0.64)	0.91 (0.53)
	CM	1.64 (0.91)	1.60 (0.93)	1.76 (1.22)	1.68 (1.10)
	CR	1.29 (0.71)	1.21 (0.70)	1.38 (0.76)	1.31 (0.75)
CM	NC	1.36 (0.64)	1.28 (0.65)	1.28 (0.86)	1.25 (0.76)
	CL	1.07 (0.54)	0.98 (0.52)	0.96 (0.66)	0.91 (0.59)
	CM	1.49 (0.91)	1.47 (0.92)	1.71 (1.24)	1.60 (1.12)
	CR	1.32 (0.78)	1.23 (0.78)	1.36 (0.79)	1.29 (0.75)
CR	NC	1.28 (0.86)	1.21 (0.84)	1.29 (0.85)	1.25 (0.73)
	CL	1.26 (0.73)	1.21 (0.80)	1.03 (0.60)	0.96 (0.55)
	CM	1.75 (1.14)	1.61 (1.19)	1.63 (1.26)	1.57 (1.13)
	CR	1.23 (0.59)	1.16 (0.52)	1.31 (0.67)	1.21 (0.68)

Table 5.18: Localization performance (carpark dataset, log 4-bit quantization)

worsen. The performance has characteristics of almost being random and circumstantial. For example, one finds exceptionally good results like the 0.34 (0.62) for cubic root 2-bit quantization for LEMON ($k = 5$) in CM/CR, as well as abysmal ones like 2.10 (1.27) for CR/CM for the same scheme using logarithmic 2-bit quantization. It appears though that such outliers become less likely as the number of bits increases. A notable characteristic of fixed quantization schemes is that they are in effect indifferent to the particular dataset, hence their "training" is purely knowing what is the minimum and maximum value of RSSI measurements from the fingerprint set to correspondingly produce the bins. As such, fixed quantization includes cases where the performance (especially for larger number of bits) can be the best when testing based on 'training" that happened on a different dataset.

The real advantage of the fixed quantization is that it eschews the need for search, and hence computation, to find an optimal quantization. Also, at high number of bits, it partitions the range of RSSI values into adequately short intervals, such that the representative values express more granular information about their distance of a tag from the pegs. With a larger set of representative values, the range of RSSI values is essentially resampled. While the best case results of this fixed quantization at high number of bits may not be the best when compared to the optimization-based results, it also does not suffer from the testing producing poor results because of optimization/training "overfitting" performed on a another dataset. Hence, depending on use case, fixed quantization may be an adequate compromise, especially if more than a few bits are used per RSSI sample.

5.2.3 Library Dataset

Recall that unlike the carpark dataset, the library dataset is very sparse. That is, from any given location, only a small number of the pegs can be "heard" over Bluetooth. As such, most of the RSSI vectors contain "null" values, or placeholder values where data is missing. As with 1-bit quantization, we also

Scheme	$E(x)$ ($\sigma(x)$)		
	local 1-bit	**local 2-bit**	**local 3-bit**
RADAR	2.04 (1.31)	1.70 (1.07)	1.70 (1.07)
LANDMARC	2.17 (1.19)	1.64 (1.01)	1.64 (1.00)
LEMON	2.30 (1.15)	1.61 (1.09)	1.60 (1.11)
LEMON ($k = 5$)	2.27 (1.13)	1.61 (1.08)	1.61 (1.08)

Table 5.19: Localization performance (library dataset, 1-bit to 3-bit quantization)

use a placeholder value of -120 dBm.

Like with the carpark dataset, there is a modest improvement going from local 1-bit to local 2-bit quantization, similar to the jump between global 1-bit and local 1-bit quantization. However, this improvement does not carry forward to local 3-bit quantization: there is no discernible change in this case.

5.3 Coverage (2-bit)

After quantization, there is a fixed number of distinct RSSI vectors. For instance, with 1-bit quantization, we can represent the quantized RSSI value for each peg using just 2 different values. In the garage car dataset, there are 10 pegs, yielding 2^{10} distinct RSSI vectors. Concretely, for b-bit quantization with n pegs, there are 2^{bn} distinct vectors.

By performing localization on every single one of these vectors, we can obtain a sort of coverage of the quantization scheme for that particular localization algorithm. Note that for any particular set of quantization boundaries, many of the RSSI vectors are practically impossible. For example, with 2-bit quantization, we would likely not see “far” for every peg, or “near” for every peg with a high-accuracy set of bounds. Still, we aim to see all the possibilities, regardless of the likelihood of each vector appearing in real-world scenarios.

Figures 5.2, 5.3, 5.4, and 5.5 are provided as the 2-bit quantization representatives corresponding to the 1-bit coverage ones shown in the previous chapter (Figures 4.3, 4.4, 4.5, and 4.6), to which they can be compared to identify the increase in coverage with more bits used, with the notable exception

of RADAR. Note also that, again, we pick the dataset with the car in the left side (CL) of the carpark area to illustrate the coverage in the presence of an obstacle (compared to no obstacle).

RADAR constructs a quadrilateral over $k = 4$ points, with the predicted location of the tag at the centre of the quadrilateral. In the carpark dataset, there are just 25 profiling points. With quantization boundaries obtained for 2-bit quantization, 238 distinct positions are possible. These are interspersed throughout the space at regular intervals. While there are 12650 possible quadrilaterals with 25 profiling points, most of these are not seen, likely because "neighbouring" profiling points are relatively close together, and distant profiling points are not neighbours anyway. When adding a car in CL configuration, the coverage is not changed by much. In contrast to RADAR,

LANDMARC uses a weighted average for predicted location. Practically, this increases the number of possible locations, particularly because the weights are not taken from a distinct set of values but are derived using the signal space distances to their neighbors. Without a car/obstacle present, the coverage shares some form of clustering around points such as the ones demonstrated by RADAR, yielding multiple points, but grouped in diffused clusters – this behavior can be witnessed, but with a less dense set of points in the 1-bit localization coverage results. Once cars are added (here, CL), the coverage shifts dramatically. Most of the coverage is distorted.

Like LANDMARC, LEMON uses a weighted average for localization. Here, we examine both $k = 4$ (like the other two algorithms) and $k = 5$, as LEMON uses $k - 1$ neighbours for its calculation – the furthest neighbour is used solely to calculate weights. With $k = 4$ the coverage is already remarkably dense with lighter coverage in the periphery, but moving to $k = 5$ results in a remarkably wide and uniform coverage. The presence of a car in CL, appears to "attract" some of the density closer to the area of the vehicle, thinning the coverage of the opposite (right) side. Still the coverage is quite good even in CL.

Scheme	Coverage (%)
RADAR (1-bit), RADAR (2-bit)	0.15-0.18
LANDMARC (1-bit), LEMON (1-bit)	0.74-0.79
LANDMARC (2-bit)	51.33-67.12
LEMON (2-bit)	74.14-87.72

Table 5.20: Relative quantized RSSI coverage performance.

5.3.1 Comparison of Coverage Results

Finally, we summarize all the coverage results across 1-bit and 2-bit quantization. The statistics across various schemes fall in clearly distinct ranges that they can be summarized by grouping them accordingly in Table 5.20. Given the maximum possible coverage due to the convex hull is 92%, the 2-bit LEMON gets close to that limit. The results of the specific variety of 2-bit LEMON were mixed within the $74.14 - 87.82\%$ range. For example, the largest coverage (87.72%) was attained by LEMON with $k = 4$, global thresholds, with the car in the right part of the area. The lowest coverage (74.14%) was attained by LEMON with $k = 5$, local thresholds, and no car present in the area. Neither a higher k, nor a more "tuned" local threshold choice, nor even the precise location of the obstacle (or the obstacle existence) appear to impact the relative coverage performance between LEMON and LANDMARC, and there is a clear advantage in favor of LEMON. RADAR's coverage inherently does not benefit from more bits, which is not the case for LANDMARC and LEMON. Keeping in mind that the quantization process's objective is to minimize the localization error of an ensemble of points, it is interesting to see that it also results in creating a (expected) more diverse coverage, and (unexpectedly) separates the performance of LANDMARC and LEMON.

Figure 5.20 summarizes the coverage for multi-bit quantization. Note that the impact of local versus global quantization and the location of car (if present) was minimal and always within the given ranges. For RADAR and LANDMARC, $k = 4$ while LEMON includes both the $k = 4$ and $k = 5$ results in the quoted range.

Scheme	$E(x)$ ($\sigma(x)$)			
	none	**linear**	**global**	**local**
RADAR	1.21 (0.45)	1.13 (0.52)	0.94 (0.52)	0.60 (0.42)
LANDMARC	1.11 (0.50)	1.05 (0.51)	0.91 (0.46)	0.51 (0.28)
LEMON	1.14 (0.64)	1.03 (0.62)	0.94 (0.47)	0.46 (0.39)
LEMON ($k = 5$)	1.10 (0.58)	1.04 (0.51)	0.94 (0.44)	0.42 (0.34)

Table 5.21: 2-bit quantization performance (carpark dataset, train:NC, test:NC).

5.4 Conclusions

We have seen the quantization produces improvement with increasing number of bits but also demonstrates diminishing returns as more bits are used. While local quantization results can produce better results than global quantization, when the training and testing environment are the same, they are not as good when the testing happens in a modified environment, whereas the global quantization appears to be more robust as it avoid this form of "overfitting". LEMON seems to perform, on balance, well, if not the best, across all cases. Using fixed quantization may be a shortcut when one wishes to avoid the computationally intense search for optimum quantization, but to get any meaningful performance one has to use more quantization bits so more bins exist, i.e., a finer grained representation of the RSSI range. Finally, LEMON appears to be the scheme that best covers in a uniform and dense manner the localizable space using as few as 2 bits per RSSI.

We offer here a summary of some the results, put together to compare and contrast schemes under two situations for 2 and 3 bits: (a) training and testing in the same set (NC), which is shown in Tables 5.21 and 5.23, and (b) training in NC and testing in CR. which is shown in Tables 5.22 and 5.24. All tables include linear quantization as the representative of fixed quantization schemes.

Scheme	$E(x)$ ($\sigma(x)$)			
	none	**linear**	**global**	**local**
RADAR	1.31 (0.59)	1.55 (0.66)	1.23 (0.60)	1.23 (0.60)
LANDMARC	1.20 (0.59)	1.50 (0.61)	1.14 (0.61)	1.14 (0.61)
LEMON	1.21 (0.62)	1.55 (0.66)	1.25 (0.64)	1.25 (0.64)
LEMON ($k = 5$)	1.15 (0.56)	1.50 (0.58)	1.20 (0.59)	1.20 (0.59)

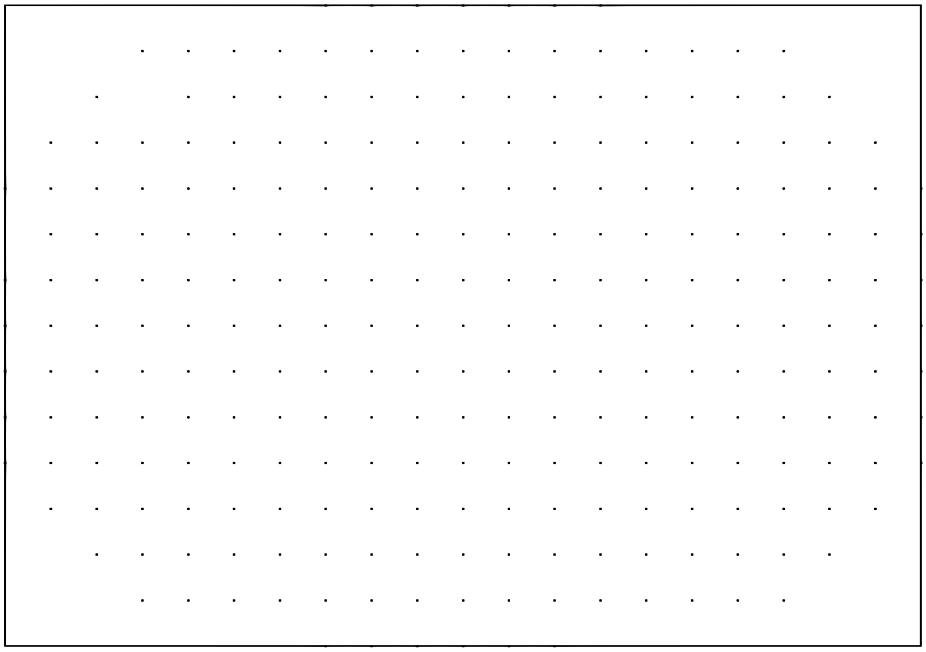
Table 5.22: 2-bit quantization performance (carpark dataset, train:NC, test:CR).

Scheme	$E(x)$ ($\sigma(x)$)			
	none	**linear**	**global**	**local**
RADAR	1.21 (0.45)	1.14 (0.64)	0.80 (0.44)	0.56 (0.42)
LANDMARC	1.11 (0.50)	1.09 (0.60)	0.76 (0.41)	0.46 (0.31)
LEMON	1.14 (0.64)	1.13 (0.61)	0.80 (0.40)	0.44 (0.39)
LEMON ($k = 5$)	1.10 (0.58)	1.12 (0.56)	0.76 (0.39)	0.41 (0.31)

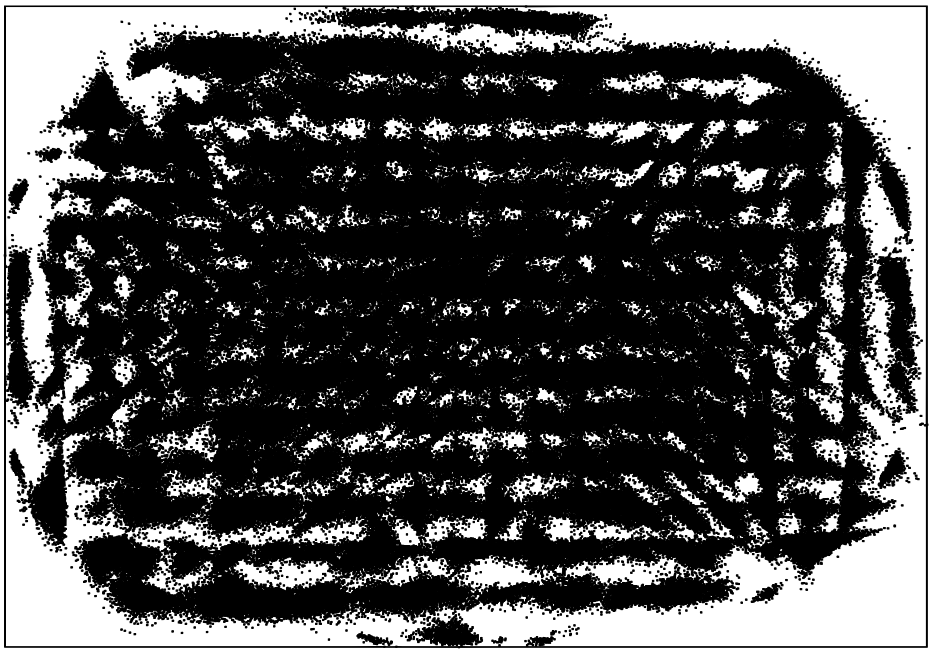
Table 5.23: 3-bit quantization performance (carpark dataset, train:NC, test:NC).

Scheme	$E(x)$ ($\sigma(x)$)			
	none	**linear**	**global**	**local**
RADAR	1.31 (0.59)	1.33 (0.59)	1.17 (0.48)	1.15 (0.49)
LANDMARC	1.20 (0.59)	1.26 (0.55)	1.06 (0.52)	1.12 (0.60)
LEMON	1.21 (0.62)	1.34 (0.58)	1.17 (0.63)	1.32 (0.70)
LEMON ($k = 5$)	1.15 (0.56)	1.28 (0.58)	1.09 (0.58)	1.20 (0.59)

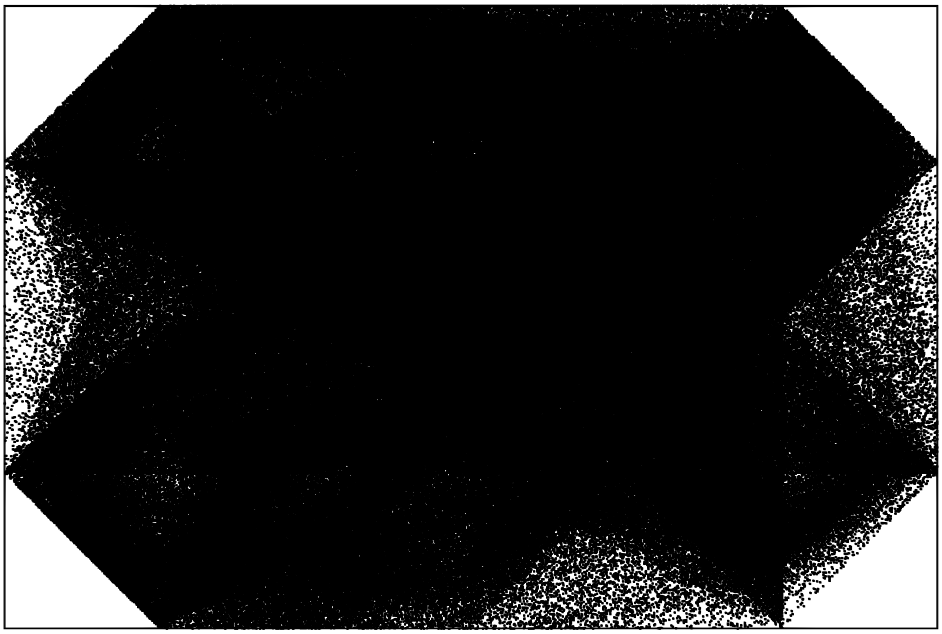
Table 5.24: 3-bit quantization performance (carpark dataset, train:NC, test:CR).



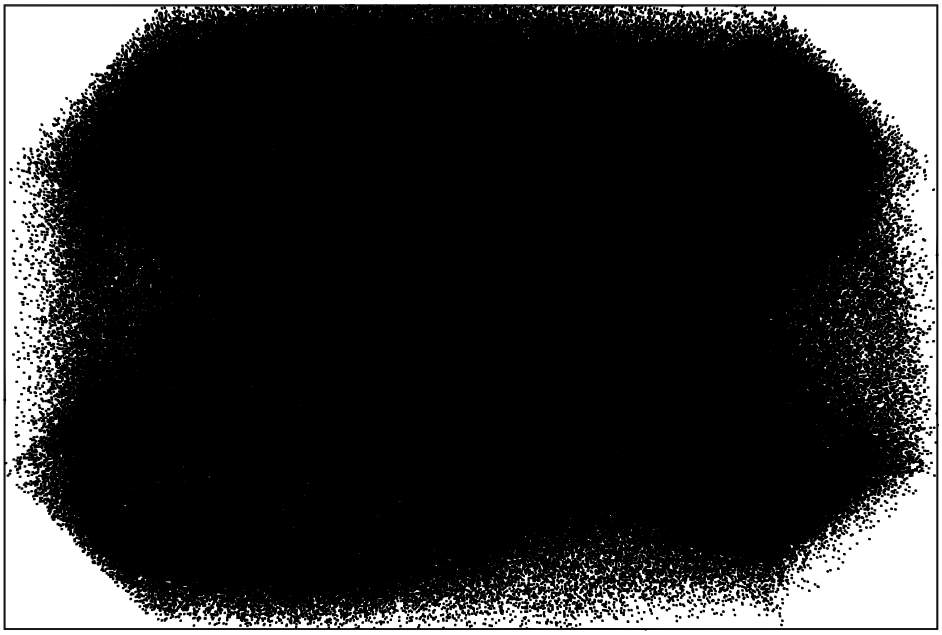
(a) RADAR



(b) LANDMARC

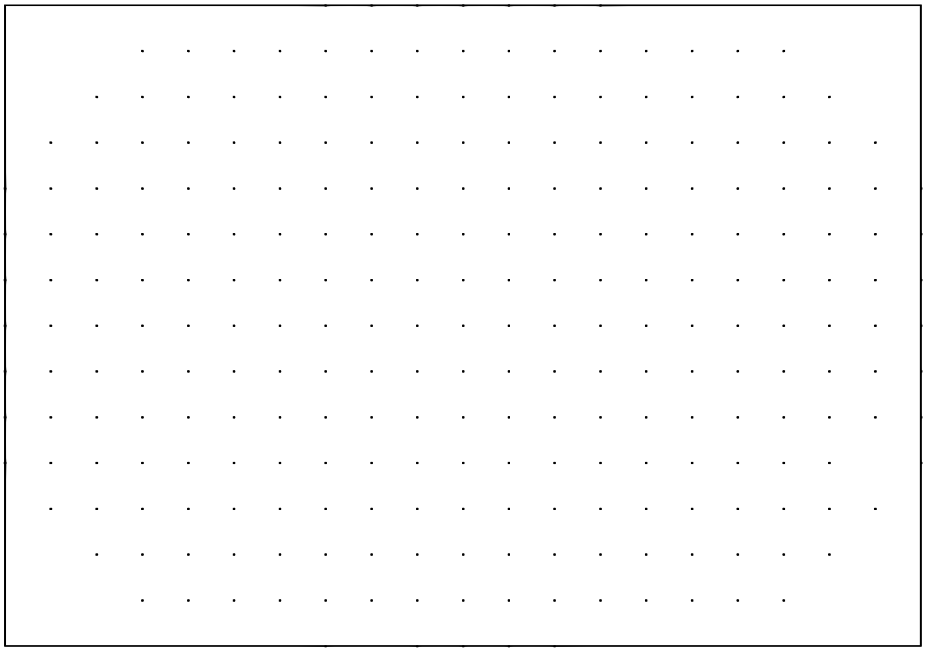


(c) LEMON

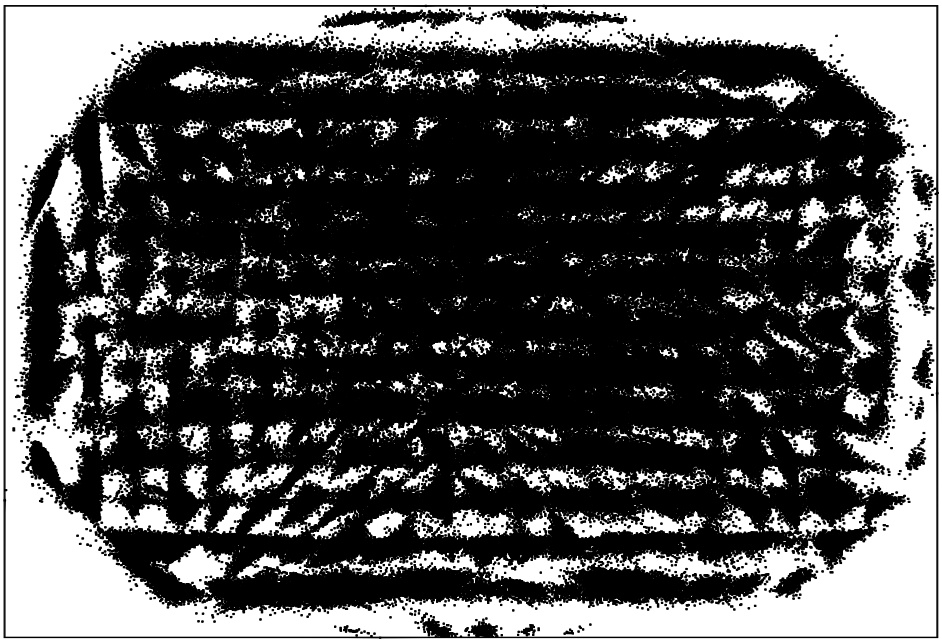


(d) LEMON ($k = 5$)

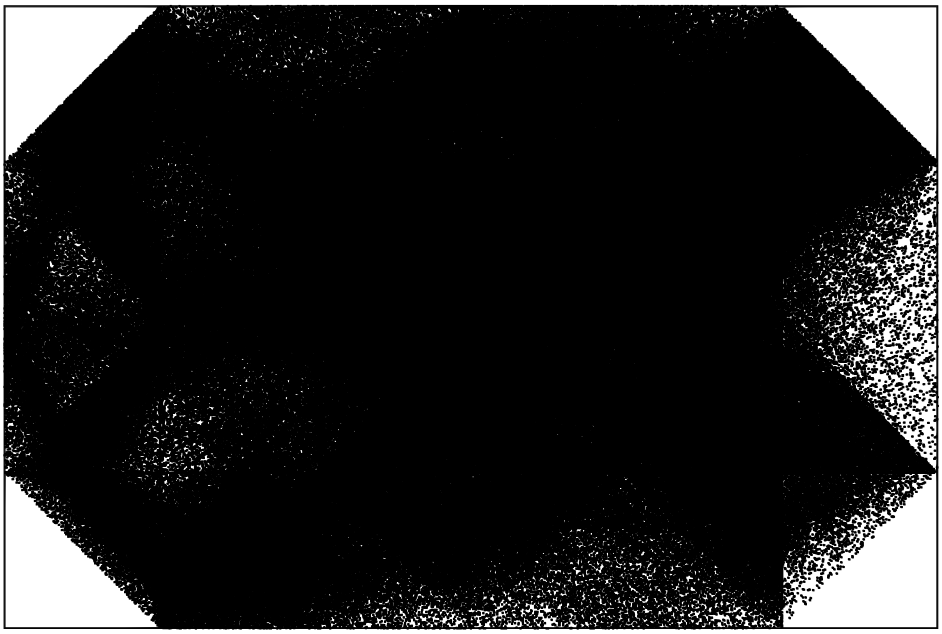
Figure 5.2: Coverage for optimal global 2-bit quantization (no car present).



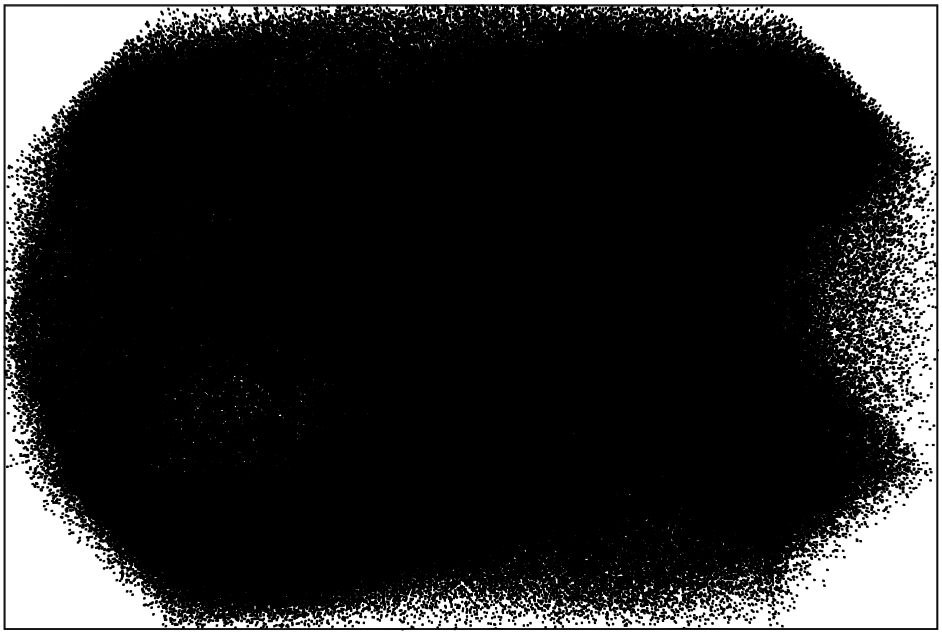
(a) RADAR



(b) LANDMARC

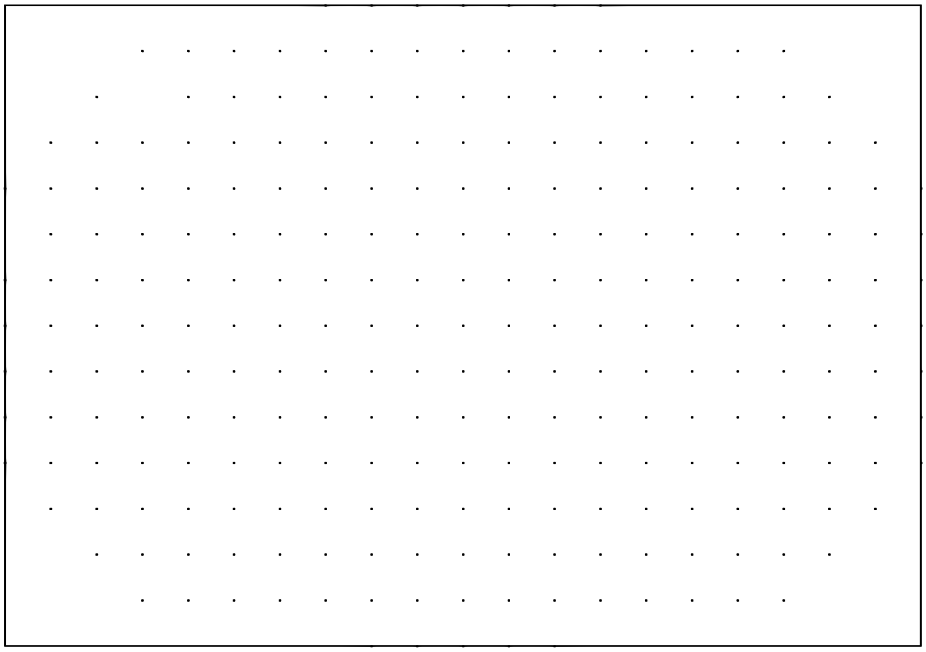


(c) LEMON

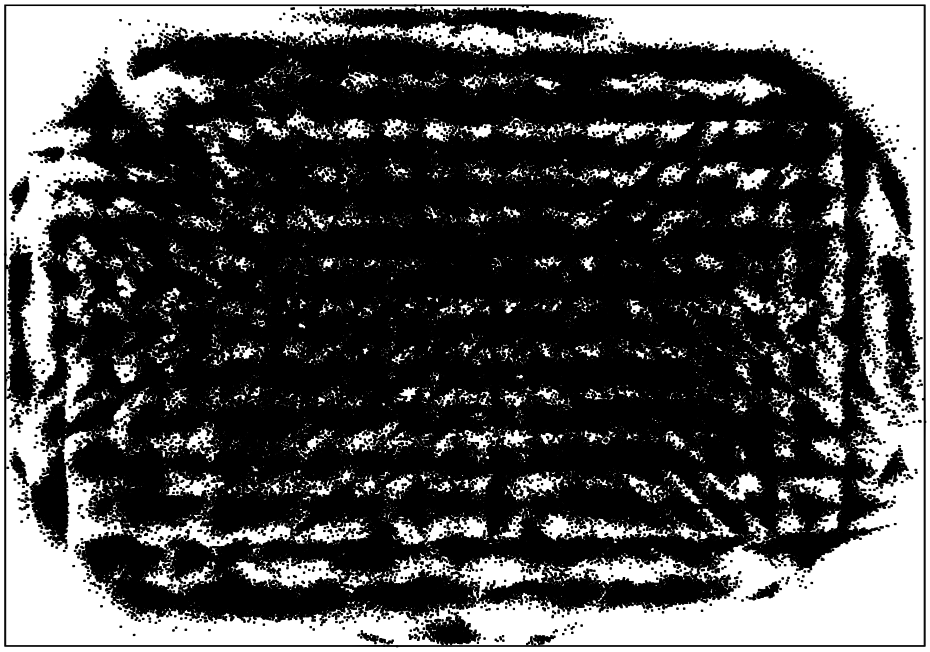


(d) LEMON ($k = 5$)

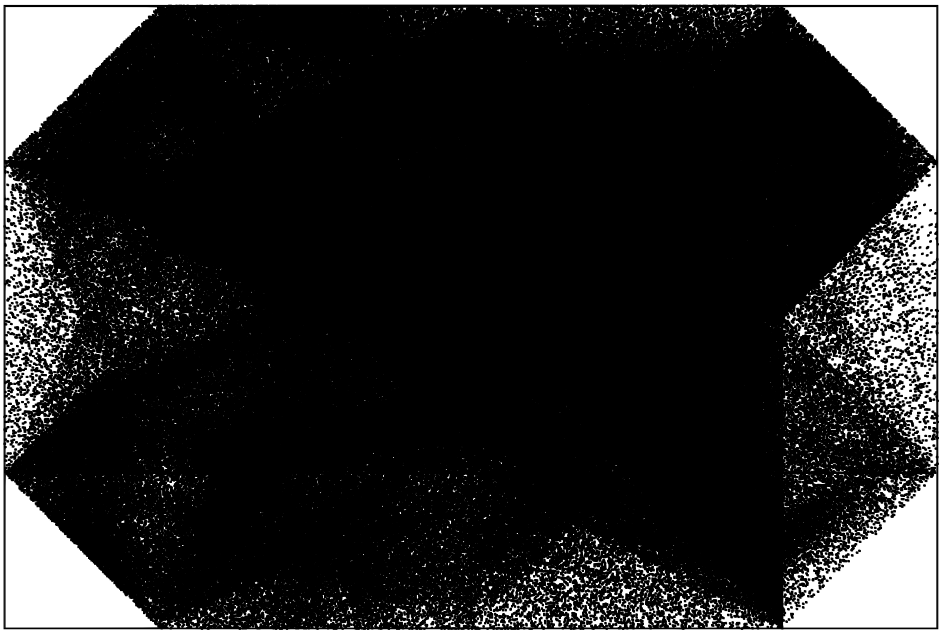
Figure 5.3: Coverage for optimal global 2-bit quantization (car present left).



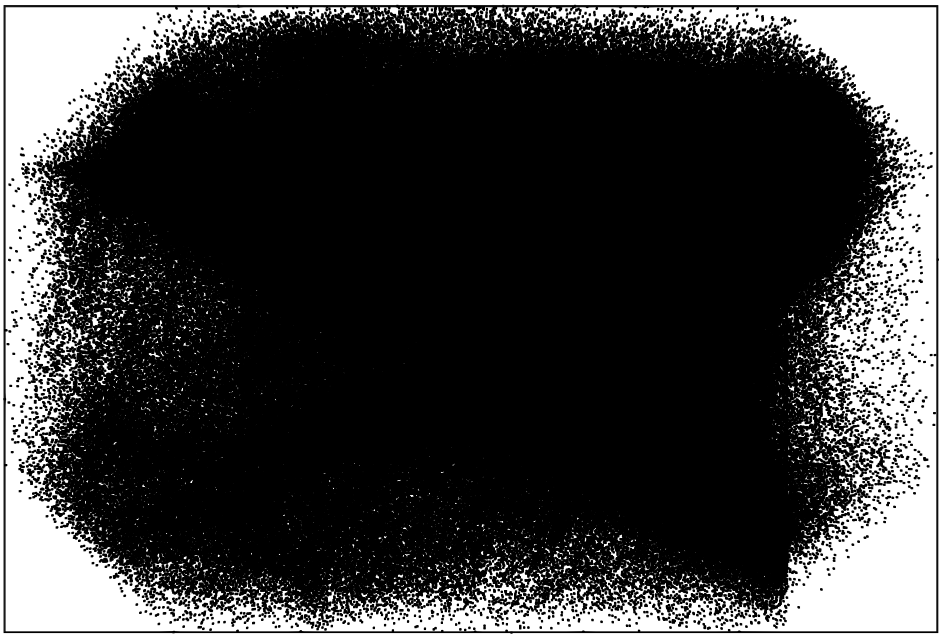
(a) RADAR



(b) LANDMARC

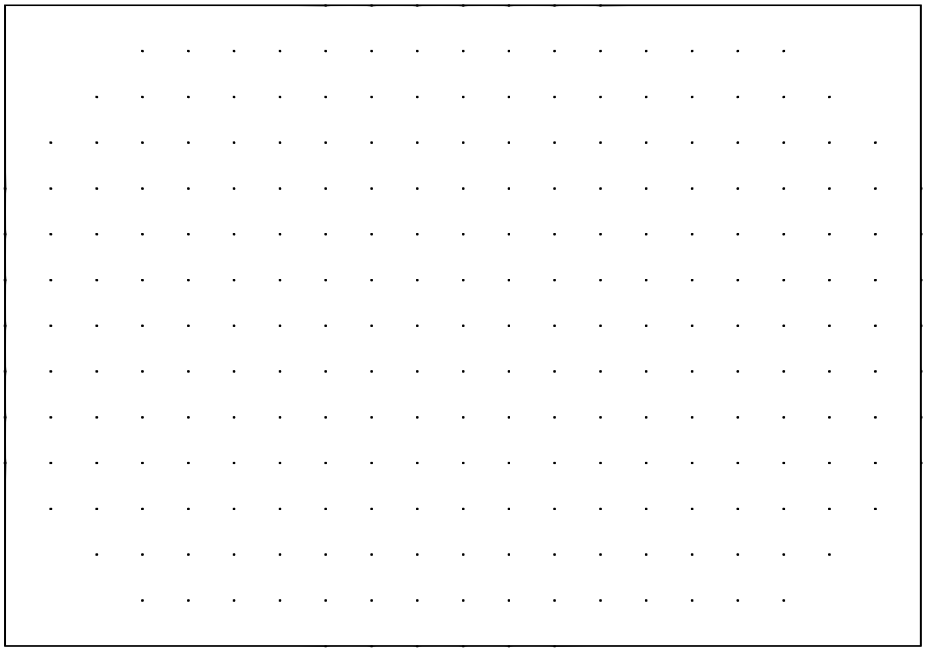


(c) LEMON

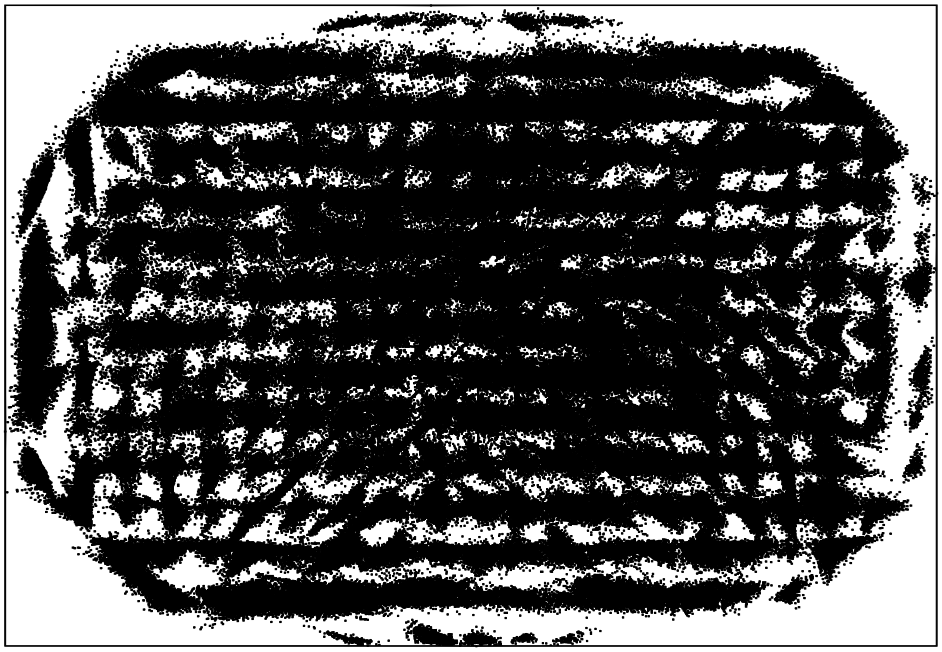


(d) LEMON ($k = 5$)

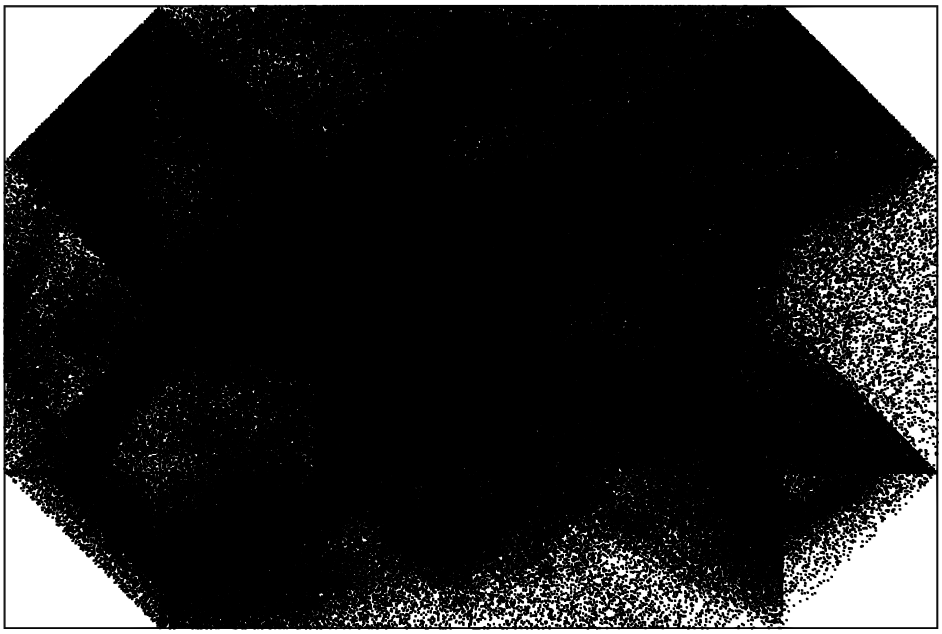
Figure 5.4: Coverage for optimal local 2-bit quantization (no car present).



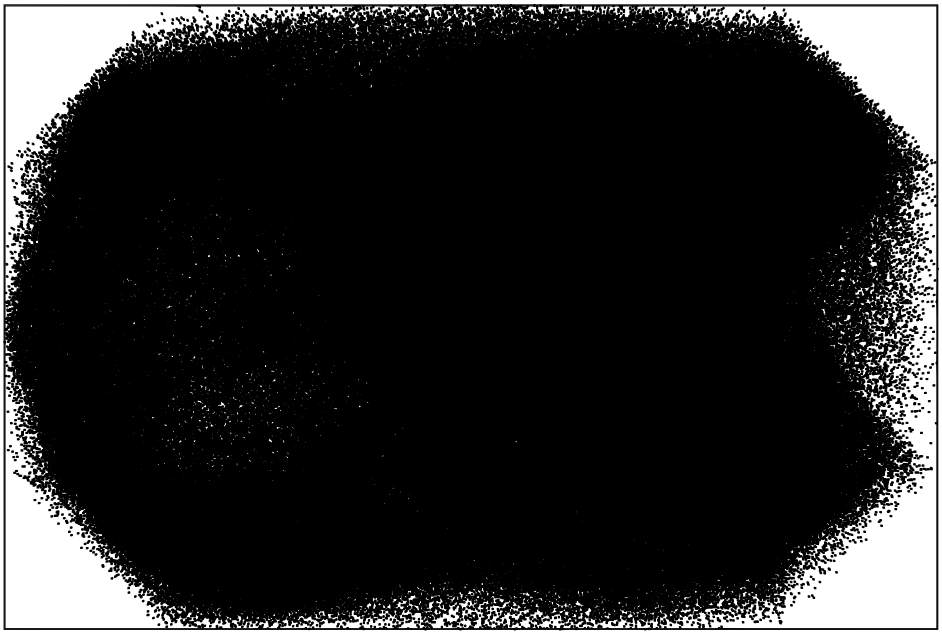
(a) RADAR



(b) LANDMARC



(c) LEMON



(d) LEMON ($k = 5$)

Figure 5.5: Coverage for optimal local 2-bit quantization (car present left).

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Through the experiments performed in previous chapters, we found that quantizing yields results on par with or sometimes better than full resolution data. The results are summarized in Figure 6.1. This plot shows $E(x)$ along with its associated $\sigma(x)$ for full-resolution data, 1-bit global and local quantization, and 2-bit global and local quantization. For each vertical line, the central dot depicts $E(x)$, with the bar showing one $\sigma(x)$ away from the mean on either side. Here, we can see that 1-bit local and 2-bit global quantization already perform similarly to full resolution data. The 1-bit quantization thresholds were obtained using EGA when not possible to perform an exhaustive search, and the 2-bit quantization thresholds were obtained using Tabu search. In Figure 6.1, the training and testing sets are the same.

Figure 6.2 is similar to Figure 6.1 but for a larger number of bits for quantization: 3-bit global and local quantization, along with 4-bit global and local quantization, comparing them also to full resolution data. Here, we can see that there is little benefit to increasing the number of bits beyond 2. While 3-bit quantization with global thresholds generally outperforms that of 2-bit global quantization, there is virtually no further improvement when going up to 4 bits. The same is true for local quantization, with 3- and 4-bit local quantization performing very similarly to 2-bit quantization.

In addition to evaluating the localization accuracy of k-NN algorithms us-

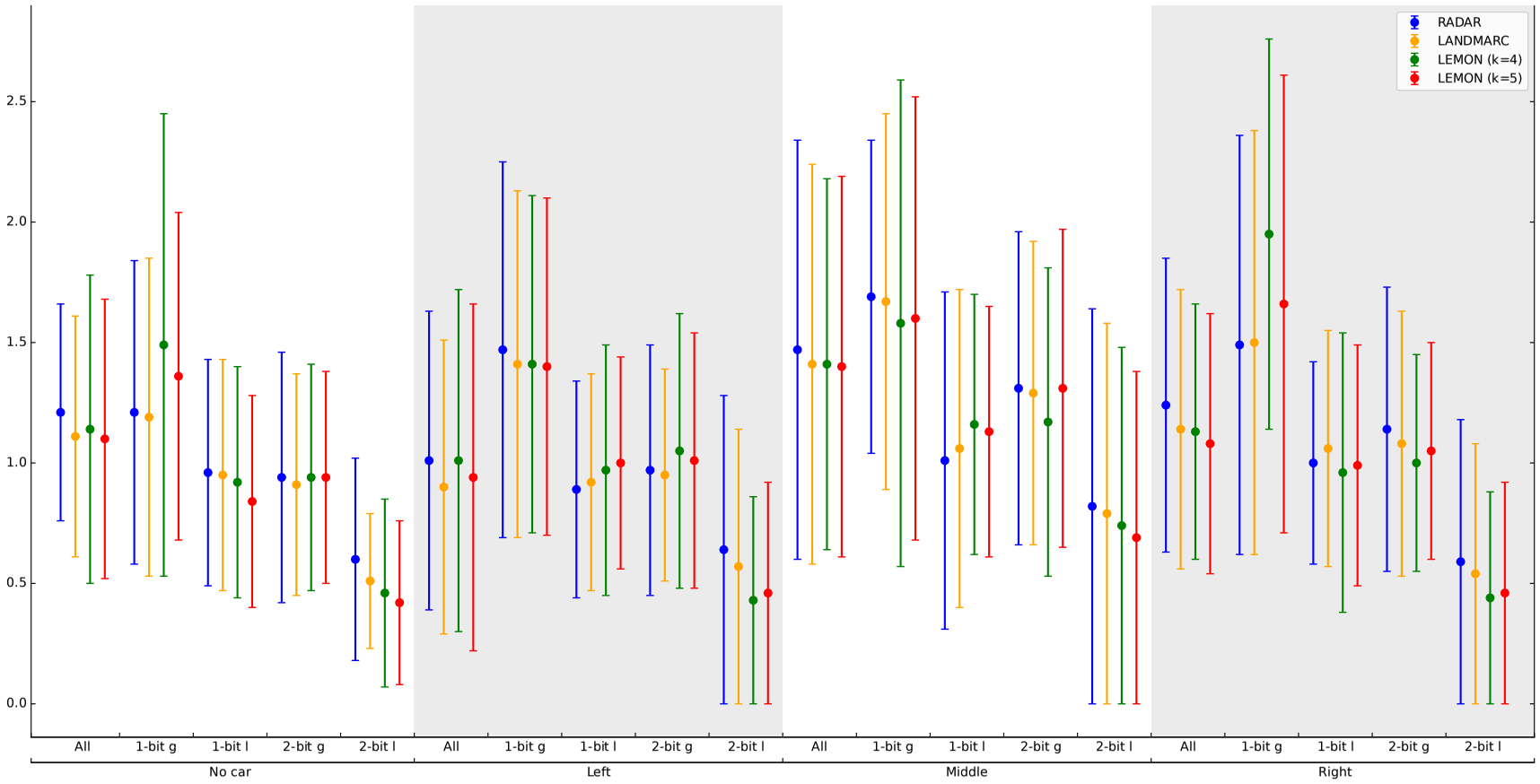


Figure 6.1: Localization performance for no quantization, 1-bit global/local and 2-bit global/local quantization.

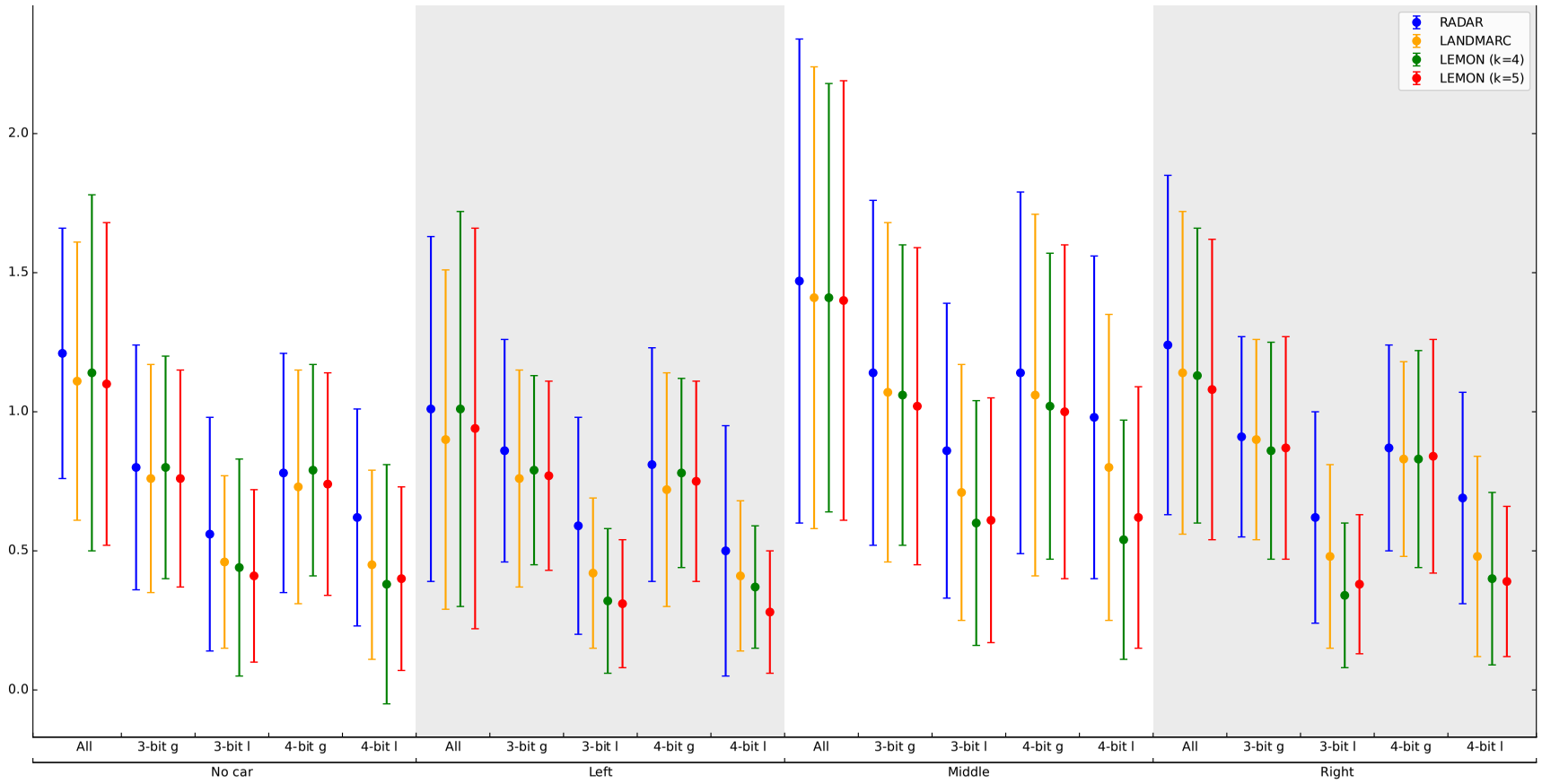


Figure 6.2: Localization performance for no quantization, 3-bit global/local and 4-bit global/local quantization.

ing quantized data, we also assessed the coverage of each. Intuitively, the coverage evaluates the "expressiveness" of each combination of quantization level and localization scheme. That is, we answer the question: which points can each localization scheme actually yield under quantization? As each algorithm estimates localization differently, we expected a difference here as well. As it uses a simple average, RADAR had the lowest expressiveness. It could only yield location estimates where each estimate was the midpoint of a polygon with points at each neighbour. As our carpark dataset fingerprints were collected at specific points on a grid, the coverage diagram generated by RADAR is similarly, points in a grid. LANDMARC and LEMON were both much more expressive, each able to localize to an incredible assortment of points. While LEMON's coverage area generally covers the entire space (bounded by the outer neighbours), LANDMARC's coverage exhibits a more clustered behavior around the points that RADAR generates, the centres of k-sized polygons. Despite differences in coverage, these seem to play little role in localization accuracy.

Lastly, we examined the effect of a sparse data with quantization. Whereas almost every point in the localization area in the carpark can receive signals from every possible peg device, this is not the case with the library, where at any given point, only a small fraction of peg devices can be detected. As with the much fuller dataset of the carpark, there is little difference across localization algorithms, even after quantization is applied. In contrast, to the carpark, however, there is little increase in mean error even with 1-bit global quantization, and 1-bit local quantization also performs similarly to using full resolution data.

6.2 Future Work

Two obvious directions for future work are (a) generalization of the performance study to 3-dimensional localization and (b) use of different "black box" localization schemes. The first is hindered by the relative paucity of datasets

that also include modifications of the 3-dimensional environment (like we had in the carpark dataset). The second option is easier to attack albeit the prevalence of machine learning schemes could result in the need to have larger datasets to use for training and evaluation.

There are results, e.g., the survey by Ahmadi and Bouallegue [1] that suggest machine learning alternatives to k-NN are capable of better performance. It would make sense to consider different "black box" localization schemes outside those of k-NN. For k-NN algorithms, the appeal of quantization lies in the ability to memorize all the localization results, or store all the results in a lookup table. This allows a given device to self-localize. However, other techniques may also yield the same benefit. In particular, localization techniques involving neural networks require very little storage as well. The output of training a neural network is simply the coefficients of the neural connections themselves; the structure of the neural network is fixed and known beforehand. Combining the coefficients and the structure, location is estimated using a mathematical formula. The devices then can also hold this formula could self-localize, as the computation is typically not complex.

We will examine three varieties of neural network-based algorithms. In each case, the neural network is trained offline, using the profiling dataset. To train each model, the profiling dataset is split into two chunks: one chunk containing 80% of the points (used for training the weights themselves), and one chunk containing the remaining 20% of the points (used to test the model derived from the training phase). To assess the accuracy of the model, the testing dataset is used, as in the other models. It is worth noting again that in the carpark dataset the profiling dataset and the testing datasets contain no overlap and could be merged into one set, and subsequently split in the 80%/20% manner suggested in random ways. Thus, the localization models are tested on points they would have never seen before during training. Neverhtless the overall carpark dataset is not particularly large and we may face limitations due to the dataset size.

We attempted (not reported in this thesis) to use a simple neural network,

using standard regression. The neural network has just one hidden layer, and the number of nodes in that layer are determined experimentally. Using profiling data set to train the neural network using back-propagation, the number of nodes is increased until no further improvement in localization accuracy is seen. The activation function for the hidden layer is ReLU, and a linear function is used for the output layer. We also tried a sigmoid activation function for the hidden layer, but the convergence time was much longer with no improvement in localization accuracy.

There might be better prospects for deep belief networks (DBN), the representatives of "deep learning" approaches that has seen good localization accuracy compared to k–NN methods, at least in the non-quantized setting. For example, Jiang et al. [13] find that the majority of points can be localized within 0.5m. LANDMARC localizes within 2m in the same setting. We aim to evaluate the DBN model using quantized RSSI vectors and determine if similar localization improvements can be obtained even if the majority of bits are dropped. Their neural network architecture is comprised of 4 hidden layers, each being a restrictive Boltzmann machine (RBM). Each layer/RBM is trained using single-stop contrastive divergence (CD-1), and finally the architecture as a whole is fine-tuned using backpropagation to minimize errors.

Interestingly, this neural network is not used *directly* to localize, but to "reconstruct" the fingerprint database. The idea here is to extract the most relevant features of RSSI data, and reconstruct them in a new fingerprint database. Therefore, the input and output of the network are both fingerprints. The DBN approach inherently recognizes that not all information in an RSSI vector is valuable.

In the online phase, the testing dataset is used as input to the DBN. The output is then compared to the reconstructed fingerprint database, and the closest matching fingerprint is used as the location. Here, the fact that our training and testing datasets have no overlap becomes important: none of the fingerprints in the database are close to the points in the evaluation dataset.

We also know a-priori, what would be the minimum error due to the locations of the testing and fingerprinting points, which for the carpark dataset is 1m. Thus, we could combine the DBN approach with our other localization schemes. Can the reconstructed fingerprints from DBN be used for example, in conjunction with k–NN or simple neural network algorithms to improve their performance, particularly with a lower number of bits?

Bibliography

[1] Ahmadi, H., Bouallegue, R.: Exploiting Machine Learning Strategies and RSSI for Localization in Wireless Sensor Networks: A Survey. In: Proc. of the 13th Intl. Wireless Communications and Mobile Computing Conf. (IWCMC). pp. 1150–1154. IEEE (Jun 2017), https://doi.org/10.1109/IWCMC.2017.7986447

[2] Bahl, P., Padmanabhan, V.N.: RADAR: An In-Building RF-based User Location and Tracking System. In: Proc. of the 19th Annual Joint Conf. of the IEEE Computer and Comm. Societies (INFOCOM). vol. 2, pp. 775–784. IEEE (Mar 2000), https://doi.org/10.1109/INFCOM.2000.832252

[3] Chowdhury, T.J., Elkin, C., Devabhaktuni, V., Rawat, D.B., Oluoch, J.: Advances on Localization Techniques for Wireless Sensor Networks: A Survey. Computer Networks 110, 284–305 (2016), https://doi.org/10.1016/j.comnet.2016.10.006

[4] Dang, X., Si, X., Hao, Z., Huang, Y.: A Novel Passive Indoor Localization Method by Fusion CSI Amplitude and Phase Information. Sensors 19(4), 875 (Jan 2019), https://doi.org/10.3390/s19040875

[5] Farid, Z., Nordin, R., Ismail, M.: Recent Advances in Wireless Indoor Localization Techniques and System. Journal of Computer Networks and Communications 2013 (2013), https://doi.org/10.1155/2013/185138

[6] Friedman, J.H., Bentley, J.L., Finkel, R.A.: An Algorithm for Finding Best Matches in Logarithmic Expected Time. ACM Transactions on

Mathematical Software 3(3), 209–226 (Sep 1977), https://doi.org/10.1145/355744.355745

[7] Gao, W., Nikolaidis, I., Harms, J.J.: RSSI Quantization for Indoor Localization Services. In: Proc. of the IEEE 28th Annual Intl. Symposium on Personal, Indoor, and Mobile Radio Comm. (PIMRC). pp. 1–7 (Oct 2017), https://doi.org/10.1109/PIMRC.2017.8292429

[8] Glover, F.: Tabu Search—Part I. ORSA Journal on Computing 1(3), 190–206 (Aug 1989), https://doi.org/10.1287/ijoc.1.3.190

[9] Haque, I.T., Nikolaidis, I., Gburzynski, P.: A Scheme for Indoor Localization through RF Profiling. In: Proc. of the 2009 IEEE Intl. Conf. Comm. (ICC) Workshops. pp. 1–5. IEEE (Jun 2009), https://doi.org/10.1109/ICCW.2009.5207992

[10] Haque, I.T., Assi, C.: Profiling-Based Indoor Localization Schemes. IEEE Systems Journal 9(1), 76–85 (2015), https://doi.org/10.1109/JSYST.2013.2281257

[11] He, S., Chan, S.H.G.: Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons. IEEE Communications Surveys & Tutorials 18(1), 466–490 (2016), https://doi.org/10.1109/COMST.2015.2464084

[12] Heurtefeux, K., Valois, F.: Is RSSI a Good Choice for Localization in Wireless Sensor Network? In: Proc. of the IEEE 26th Intl. Conf. on Advanced Information Networking and Applications (AINA). pp. 732–739. IEEE (Mar 2012), https://doi.org/10.1109/AINA.2012.19

[13] Jiang, H., Peng, C., Sun, J.: Deep Belief Network for Fingerprinting-Based RFID Indoor Localization. In: Proc. of the 2019 IEEE Intl. Conf. in Comm. (ICC). pp. 1–5 (May 2019), https://doi.org/10.1109/ICC.2019.8761800

[14] Joaqun Torres-Sospedra, R.M.: UJIIndoorLoc (2014), https://archive.ics.uci.edu/dataset/310

[15] Kaddouri, S., Hajj, M.E., Zaharia, G., Zein, G.E.: Indoor Path Loss Measurements and Modeling in an Open-Space Office at 2.4 GHz and 5.8 GHz in the Presence of People. In: Proc. of the IEEE 29th Annual Intl. Symposium on Personal, Indoor and Mobile Radio Comm. (PIMRC). pp. 1–7 (Sep 2018), https://doi.org/10.1109/PIMRC.2018.8580695

[16] Khandker, S., Torres-Sospedra, J., Ristaniemi, T.: Analysis of Received Signal Strength Quantization in Fingerprinting Localization. Sensors 20(11) (Jun 2020), https://doi.org/10.3390/s20113203

[17] Liu, H., Darabi, H., Banerjee, P., Liu, J.: Survey of Wireless Indoor Positioning Techniques and Systems. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews) 37(6), 1067–1080 (Nov 2007), https://doi.org/10.1109/TSMCC.2007.905750

[18] Mehdi Mohammadi, A.A.F.: BLE RSSI Dataset for Indoor localization and Navigation (2017), https://archive.ics.uci.edu/dataset/435

[19] Mizmizi, M., Reggiani, L.: Design of RSSI Based Fingerprinting with Reduced Quantization Measures. In: Proc. of the Intl. Conf. on Indoor Positioning and Indoor Navigation (IPIN). pp. 1–6. IEEE (Oct 2016), https://doi.org/10.1109/IPIN.2016.7743606

[20] Mohammadi, M., Al-Fuqaha, A., Guizani, M., Oh, J.S.: Semisupervised Deep Reinforcement Learning in Support of IoT and Smart City Services. IEEE Internet of Things Journal 5(2), 624–635 (Apr 2018), https://doi.org/10.1109/JIOT.2017.2712560

[21] Ni, L.M., Liu, Y., Lau, Y.C., Patil, A.P.: LANDMARC: Indoor Location Sensing Using Active RFID. Wireless Networks 10(6), 701–710 (2004), https://doi.org/10.1023/B:WINE.0000044029.06344.dd

[22] Patwari, N., Hero, III, A.O.: Using Proximity and Quantized RSS for Sensor Localization in Wireless Networks. In: Proc. of the 2nd ACM Intl. Conf. on Wireless Sensor Networks and Applications (WSNA). pp. 20–29. WSNA '03, ACM (Sep 2003), https://doi.org/10.1145/941350.941354

[23] Popleteev, A.: AmbiLoc: A Year-Long Dataset of FM, TV and GSM Fingerprints for Ambient Indoor Localization. In: Proc. of the 8th Intl. Conf. on Indoor Positioning and Indoor Navigation (IPIN) (Sep 2017), https://orbilu.uni.lu/handle/10993/32587

[24] Reeves, C.R.: Genetic Algorithms. In: Gendreau, M., Potvin, J.Y. (eds.) Handbook of Metaheuristics, pp. 109–139. International Series in Operations Research & Management Science, Springer (2010), https://doi.org/10.1007/978-1-4419-1665-5_5

[25] TI: CC1100 Data Sheet, https://www.ti.com/product/CC1100

[26] Wu, K., Xiao, J., Yi, Y., Chen, D., Luo, X., Ni, L.M.: CSI-Based Indoor Localization. IEEE Transactions on Parallel and Distributed Systems 24(7), 1300–1309 (Jul 2013), https://doi.org/10.1109/TPDS.2012.214

[27] Yang, Z., Zhou, Z., Liu, Y.: From RSSI to CSI: Indoor Localization via Channel Response. ACM Computing Surveys 46(2), 1–32 (Nov 2013), https://doi.org/10.1145/2543581.2543592

[28] Yianilos, P.N.: Data Structures and Algorithms for Nearest Neighbor Search in General Metric Spaces. In: Proc. of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). pp. 311–321. SODA '93, SIAM, USA (Jan 1993)

[29] Yong, A., Nikolaidis, I., Harms, J.J.: Localization Sensitivity Under RSSI Quantization. In: Proc. of the 2019 IEEE Int. Conf. on Comm. (ICC). pp. 1–6 (May 2019), https://doi.org/10.1109/ICC.2019.8761085

[30] Zanca, G., Zorzi, F., Zanella, A., Zorzi, M.: Experimental Comparison of RSSI-Based Localization Algorithms for Indoor Wireless Sensor Networks. In: Proc. of the Workshop on Real-World Wireless Sensor Networks (REALWSN). pp. 1–5. ACM (Apr 2008), https://doi.org/10.1145/1435473.1435475

[31] Zou, H., Huang, B., Lu, X., Jiang, H., Xie, L.: Standardizing Location Fingerprints Across Heterogeneous Mobile Devices for Indoor Localization. In: Proc. of the IEEE 2016 Wireless Comm. and Networking Conf. (WCNC). pp. 1–6. IEEE (Apr 2016), https://doi.org/10.1109/WCNC.2016.7564800