SELF-CALIBRATING LOCALISATION SYSTEMS IN WIRELESS NETWORKS

An der Fakultät für Elektrotechnik
der Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg
zur Erlangung des akademischen Grades eines Doktor-Ingenieurs
vorgelegte

DISSERTATION
von

Bruno Betoni Parodi

aus Rio de Janeiro, Brasilien

Hamburg 2009
Erstgutachter: Univ.-Prof. Dr.-Ing. Joachim Horn
Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg
Professur für Regelungstechnik

Zweitgutachter: Univ.-Prof. Dr.-Ing. Gerd Scholl
Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg
Professur für Elektrische Messtechnik

Vorsitzender: Univ.-Prof. Dr.-Ing. Stefan Dickmann
Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg
Professur für Grundlagen der Elektrotechnik

Tag der mündliche Prüfung: 26.03.2010
Para Thereza, Nelcy e Fidelina, minha 3ª vó.
Acknowledgments

To Henning, Andrei, and Prof. Horn, who guided me through the world of localisation systems, and gave me the chance to develop my work.

To Pedro, Marian, Hui, Joachim, Dragan, and all colleagues from CT IC4, specially Prof. Schürmann and Thomas, for whose support to my research and to several conferences I am eternally thankful.

To my family from a distance, mom, dad, Jotalhão, Regina, Danilo, Babica, Fábio and to my family next door, Dani, Zé, Dinka, Ca, Ali, Cris, Jeann, Fê, Mari, Katarina and all those who brought balance to the force when all my faith and belief were again challenged.

To all those who perished during this long run, I wish you could have stayed longer to enjoy with me the epilogue of this chapter.

And specially to the savage Canadian O’Flynn for the monstrous help on the grammar revision.
Contents

List of Figures xi
List of Tables xiii
Preface xv

1 Introduction 1
  1.1 Motivation .................................................. 1
  1.2 Context of Indoor Radio Localisation ..................... 2
    1.2.1 Channel Modelling .................................... 2
    1.2.2 Feature Estimation and Prediction .................... 3
    1.2.3 Indoor Localisation .................................... 3
    1.2.4 SLAM .................................................. 5
  1.3 Thesis Contribution and Structure ........................ 5

2 Localisation Techniques using Radio 7
  2.1 Radio Signal Features ..................................... 7
    2.1.1 Field Strength ....................................... 7
    2.1.2 Time ............................................... 8
    2.1.3 Angle ............................................. 9
  2.2 Geometric Methods ......................................... 9
    2.2.1 Triangulation ....................................... 9
    2.2.2 Trilateration ...................................... 10
    2.2.3 Multilateration .................................... 11
  2.3 Pattern Matching .......................................... 13
    2.3.1 Nearest Neighbours .................................. 14
    2.3.2 k-Nearest Neighbours ................................ 15
    2.3.3 Maximum Likelihood .................................. 15
    2.3.4 Minimum Mean Square Error .......................... 17
  2.4 Practical Issues .......................................... 17
    2.4.1 Free Field - Line of Sight .......................... 18
    2.4.2 Non Line of Sight ................................... 18
    2.4.3 Structured Environment - Multi Path ................ 19
    2.4.4 Measurement ........................................ 21
List of Figures

2.1 Triangulation ..................................................... 10
2.2 Trilateration ....................................................... 11
2.3 Multilateration ...................................................... 12

3.1 Radial Model: (a) linear and (b) logarithmic profile ................. 27
3.2 Possible paths between a transmitter and a receiver ................ 29
3.3 Weighting function $f_{c,k}$ (a) with $f_{c,k} \in \mathbb{R}$ and (b) $f_{c,k} \in \mathbb{R}^2$ .................... 30
3.4 Convergence of $F_k$ to 1 (a) with $F_k \in \mathbb{R}$ and (b) $F_k \in \mathbb{R}^2$ .................... 35
3.5 Limit area for a linear (a) and logarithmic model (b) ................. 40
3.6 Limit volume for a linear model ................................ 41
3.7 Exp. 3.1 - Feature map learning for sequentially ordered measurements.
   The distance between two succeeding measurement positions is smaller
   than $\phi$ ............................................................... 45
3.8 Exp. 3.2 - Feature map learning for sequentially ordered measurements.
   The distance between two successive measurement positions is equal to $\phi$ 46
3.9 Exp. 3.3 - Feature map learning with uniform power distribution of the
   measurements ....................................................... 47
3.10 Exp. 3.4 - Feature map learning for randomly ordered measurements.
   The initialisation shows an offset and the measurements are noisy ... 48
3.11 Exp. 3.5 - Feature map learning with wall discontinuity ............ 49
3.12 Exp. 3.6 - Feature map learning for real world data ................. 50
3.13 Examples of self organised states with 2D lattice .................. 55
3.14 Office environment with DECT BSs ................................ 58
3.15 DECT office plant (a) localisation error and (b) feature map error 59
3.16 Office environment with WLAN BSs .............................. 59
3.17 WLAN office environment (a) localisation error and (b) feature map error 60

4.1 1D simulation - initialisation ................................... 71
4.2 1D unimodal simulation - true model ............................ 72
4.3 1D unimodal simulation - learned pdf at $k = 4200$ ................ 72
4.4 1D unimodal simulation - pdf error ............................ 73
4.5 1D unimodal simulation - position error ........................ 73
4.6 1D bimodal simulation - true model ............................ 74
4.7 1D bimodal simulation - learned pdf at $k = 4200$ ................ 74
4.8 1D bimodal simulation - pdf error ............................ 75
List of Tables

3.1 SLL Simulation Characteristics ........................................ 51
3.2 SLL Real World Experiment Results ................................. 60

4.1 SPLL Simulated 1D Experiments ...................................... 88
4.2 SPLL Simulated 2D Experiments ...................................... 90
4.3 SPLL Real World Experiment Results ............................... 95
RIKER
Computer – locate Captain Picard.

COMPUTER
The captain is not on the ship.

*From Star Trek: The Next Generation “Q Who?”*
Chapter 1

Introduction

Indoor localisation is still a challenging subject among the localisation tasks. While for outdoor environments the Global Positioning System or GPS (and soon also Galileo) is the well established and popular solution, for indoor environments there are yet many different systems and solutions, each of them with promising features and some drawbacks. GPS is satellite based and uses propagation time of radio signals. A location estimate is retrieved using signals from four or more satellites in line-of-sight (LOS), which explains why it fails to operate with good accuracy in indoor environments, where the complex setups of corridors, floors, walls, and doors attenuate and deflect the satellite signals beyond the capability of these systems to adjust themselves.

1.1 Motivation

Many indoor systems achieve an accuracy of a few metres, which is comparable to the best results achieved with GPS for outdoor scenarios. However, these systems usually need many proprietary sensors populating the area where a user must be located, as in [83] and [75], or they need many received signal strength (RSS) measurements prior to system start to build a radio map, as in [6] and [5]. The major drawbacks in such cases are the low scalability and high implementation costs. Propagation time is also used in some indoor systems, where it achieves comparable performance of systems using RSS, but using proprietary sensors and under additional assumptions, as for example LOS, that limit generalisations as in [78] and [79].

The main advantage of using existent wireless networks for localisation is that no
extra hardware is needed to be installed (as is the case with special sensors and tag-based systems) since the RSS measurement is a standard feature of wireless communication systems. However, this advantage is opposed by the costly calibration phase that must be accomplished before the system start. In order to build the RSS measurement database that such localisation systems require, technicians must collect samples in the area where localisation must be performed. These samples are recorded with the information of where exactly on the map they were taken and sometimes with a time stamp too, being often referred to as labelled samples. This process of going around the whole area, marking manually where the measurement was taken, and measuring the RSS may be extremely time consuming if the locating area comprises a campus with many facilities and multi-storey buildings. In such cases, this process, also known as fingerprinting, may take months and requires a team of technicians dedicated only to this task.

This work aims at the reduction of this calibration effort in localisation systems based on wireless networks.

1.2  Context of Indoor Radio Localisation

The field of radio localisation has spread through a very diverse spectrum of areas, going from general aspects of physics and systems, to very specific solutions with proprietary devices. Major subjects that were particularly important or even interesting to the development of this work are as follows.

1.2.1  Channel Modelling

Before the applications for radio localisation were even sketched, many works dealt with radio channel modelling. Most of these works assumed radial propagation, not necessarily with LOS but in a homogenous space, which made them unsuitable for indoor locations in structured environments. They usually concentrated their efforts in a very specific radio frequency and with raw radio pulse signals to determine the impulse response. Nevertheless, they contributed strongly to the understanding of radio propagation behaviour and the meaning of the extracted radio features providing
a start for localisation systems.

The famous work of Valenzuela [91] from AT&T Bell Labs was one of the first to address both multipath and statistical modelling in indoor environments. The progression from deterministic equations from physics to probability models that describe signal fading and phase shifting was repeated by many authors using different frequencies and statistical models as in [3,22,23,49,50,84]. A curious work in [44] models the human body disturbance in the radio propagation as a water cylinder.

1.2.2 Feature Estimation and Prediction

**RSS:** Aside from the usual radial propagation model (as in section 2.1.1), another method to estimate the RSS of radio signals uses ray tracing, as presented in [42,96,103,104]. Some variations of ray tracing receive other names such as beam tracing [37,38], or ray splitting [63]. This technique is very accurate for estimates where multipath occurs. However, this technique is very computationally expensive and requires a precise geometric and physical description of the spatial environment. Between the simplicity of radial modelling and the precision of ray tracing is a method called dominant path, explained in [97,102] (see section 3.2.1.2).

**Time:** The time measurement in multipath environments was also the focus of several groups that pursued localisation based on time. In [69,110] the time of arrival (ToA) measurement is addressed and in [81] the delay estimation (see section 2.1.2). The bandwidth influence on distance error for localisation is treated in [1,2].

**Angle:** The measurement of the Angle of Arrival (AoA) is more complex to achieve, and is commonly presented with time estimation as the angle is usually estimated from small delays in an antenna array (see section 2.1.3). In [58,94,95] the estimation of AoA is issued. A similar approach to determination of Direction of Arrival (DoA) is given in [43].

1.2.3 Indoor Localisation

Many of the constantly referenced early works on indoor location do not deal exclusively with RSS or even radio signals. In these cases, it is common to find proprietary devices
dedicated exclusively to the localisation task.

In [98], for example, the Active Badge presents a solution based on infrared (IR) pulses from a mobile device to fixed sensors at known positions. In [99] the mobile radio triggered devices emit ultrasonic pulses to fixed sensors instead, which permit trilateration using ToA. In [48] the mobile device was miniaturised and called Bat unit. The Cricket Location Support System, presented in [83], also uses ultrasonic mobile devices triggered by radio signals.

The RADAR system [5, 6], from Microsoft Research, was one of the first to use exclusively RSS to locate users using the already available infrastructure of WLAN. Likewise, the MagicMap [57] and the Palantir [40] also use RSS and WLAN.

The SpotON [52, 56] and the LANDMARC [75, 76] also use RSS, but from RFID tags instead of from WLAN.

Surveys containing resumes of the most referenced works with different comparisons between them can be found at [53, 54, 70, 79, 80].

So far, the papers using RSS and WLAN apply pattern matching approaches like the nearest neighbours (NN) to retrieve a user location based on previously measured calibration data (see section 2.3.1). In [27] linear regression is used to fit measurement sets based on radial models and using wall attenuation factors. Probabilistic approaches leading to Bayesian localisation appeared later then.

In [24] a Bayesian network is used to retrieve room localisation. A research group from Helsinki Institute for Information Technology presents Bayesian localisation solutions in [61, 62, 86, 87], also using calibration data. Another research group from University of Trento also dealt with Bayesian localisation in [9, 10, 21], as well as researchers from University of Maryland in [106–109].

From the same group that earlier developed the SpotON, in [39, 55] there are interesting results with Bayesian filtering techniques as particle filters for user localisation refinement and tracking. In [41] particle filters are also used to retrieve location using Markov chain modelling.

An interesting but simplistic way of improving the localisation system using new measurements is found in [25, 26].

User tracking and sensor fusion from WLAN and Bluetooth are explored in [45] and in [64–66] tracking using WLAN is applied to locate a robot, which can be seen
as an introduction to SLAM.

1.2.4 SLAM

Simultaneous Localisation and Mapping (SLAM) deals with the task of locating a robot and building a map at the same time. Geometric information retrieved from the robot movement models are associated with distance measurements from remote sensors using either RF, IR, ultrasound or image processing, for example.

This subject is also vast and is here mentioned to point out that the algorithms developed in this thesis, the Simultaneous Localisation and Learning (SLL) and the Simultaneous Probabilistic Localisation and Learning (SPLL), are not related to SLAM, as their names may suggest. The fundamental difference is that in SLAM the geometric map is unknown, later being built from the measurements, fixed landmarks and movement models, while in SLL/SPLL the geometric map is given, and the feature information contained within is fetched from the measurements.

Works about SLAM can be found in [4, 28, 46, 47, 67, 73, 77, 85, 89, 101].

1.3 Thesis Contribution and Structure

This thesis introduces two novel methods for localisation systems.

The first, introduced in Chapter 3, is the Simultaneous Localisation and Learning (SLL). This method addresses the reduction of the calibration effort in RSS based systems. The SLL uses a rough initial model to start the system, with very little information, such as the base station (BS) positions. The initial accuracy is equivalent to Cell-ID systems and, through successive localisation queries, combined with an iterative learning algorithm, achieves an accuracy equivalent to systems that use calibration measurements, but without the effort of collecting these measurements. The system uses only unlabelled samples, that is, plain RSS measurements without information about their true location, contrasting with the required labelled samples for usual calibration.

The SLL is based on the Self Organising Maps or SOMs and records as a feature the mean value of RSS in the feature map, performing localisation with a pattern matching
approach called nearest neighbours (NN).

The second method, presented in Chapter 4, is a natural evolution from SLL, called Simultaneous Probabilistic Localisation and Learning (SPLL). This method records as a feature the probabilistic distribution of RSS in the feature map. The localisation is performed accordingly with a probabilistic inference method called minimum mean squared error (MMSE), instead of pattern matching. Also, the learning uses non-parametric density estimation to update the feature map. As the SLL is based on SOMs, the same could be said of SPLL and the Bayesian SOM (BSOM), as in [105].

The major characteristic of both SLL and SPLL is the use of unlabelled samples as a base for the calibration (usually done with costly labelled samples). The learning phase of both algorithms can work continuously while the system is operational such that changes in the environment can also be learned after the system start.

This work is the result of three years of research at Siemens AG, CT IC-4 in Munich, Germany. Its evolution can be tracked by the following publications from the author of this thesis. In [16] the SLL was first proposed, yet without a specific name. In [13,14,68] some important properties and conditions for the successful use of SLL were proven. A thorough comparison between SLL and SOM was given in [15]. In [17,18] the SPLL was introduced and in [12,19] a generalisation for 2D space was given. A patent for the SLL was filed in [8] and for the SPLL in [7].

This thesis is structured as follows.

In chapter 2 the fundamental issues related to the general localisation task are addressed.

In chapter 3 the SLL method is introduced. Its main characteristics, analytical proofs, and working conditions are presented together with theoretical examples. Real world experiments validate the algorithm using WLAN and DECT networks.

In chapter 4 the SPLL method is introduced. The original SLL algorithm is extended to use probabilistic localisation and learn distributions as features. Theoretical examples are given showing its capabilities. Real world experiments using a WLAN network validate the algorithm.

Chapter 5 concludes this work and makes relevant suggestions for future development.
Chapter 2

Localisation Techniques using Radio

The localisation task using radio signals can be stated as: retrieve a user (or a mobile terminal or a tag) location in a reference frame (or map) using features obtained from radio signals.

2.1 Radio Signal Features

A radio system comprises one or more transmitting devices, each one equipped with a suitable antenna, a propagation medium, which in most cases is the air, and one or more receivers. Both transmitters and receivers may be fixed in space or mobile. Their location may be known or not.

If the radio system is used in a cellular topology, then a transmitting device is also known as Base Station (BS) or Access Point (AP). The receivers are usually mobile devices and also capable of transmission.

The electromagnetic waves spread by the transmitters possess many characteristics inherent to the communication system itself, such as frequency, modulation type, transmitting power, bandwidth, and so on. The most common features extracted from radio signals and used for localisation purposes are listed as follows.

2.1.1 Field Strength

The power of the electromagnetic field emitted by a transmitter spreads through space and decays with increasing distance from the transmitting antenna. In free space, the
propagation can be modelled as a linear decay of the received power with increasing distance on a logarithmic scale:

\[ p_r = p_{\text{ref}} - 10 \cdot \gamma \cdot \log \left( \frac{d}{d_{\text{ref}}} \right), \]  

(2.1)

where \( p_r \) is the received power in dBm as a function of the distance \( d \), and \( p_{\text{ref}} \) is a constant term representing the received power in dBm at a reference distance \( d_{\text{ref}} \). The model parameter \( \gamma \) is equal to 2 in free space. In indoor environments \( \gamma \) may assume values between 2 and 6 [84].

Hence the feature \( p_r \), usually referred to as signal strength (SS) or received signal strength (RSS), can be transformed into spatial information needed for localisation.

### 2.1.2 Time

The time elapsed between the emission of a radio wave and its reception by a mobile terminal can also be used to retrieve spatial information. As the speed of electromagnetic waves is constant for some known medium (\( c \) at the vacuum), calculating the distance travelled by a wave in free space is rather trivial.

When only the time between a transmission and reception is considered then the term Time of Arrival (ToA) is used. An accurate synchronisation between the transmitter and receiver is required in order to use the ToA.

With two or more transmitters the time difference between the received signals can be used rather than the absolute time of flight of each individual signal. In this case the term Time Difference of Arrival (TDoA) is used. Here all transmitters must be synchronised in order to use TDoA, whereas the receiver need not be synchronised with anyone else.

If the receiver is also capable of transmitting, then the starting transmitter can wait for an acknowledgement signal from the receiver, counting the whole time elapsed. In this case the term Round Trip Time is used. The synchronisation requirement can be relaxed for this case, but the time required by the receiver to generate the acknowledgement must be known.
2.1.3 Angle

The Angle of Arrival (AoA) is the only feature that has a direct geometric meaning. It retrieves a receiver position using trigonometry. However, measuring the AoA is usually more difficult than measuring signal strength or time. The hardware required to measure AoA is an antenna array, where each single antenna from the array has a distinct known orientation. The differences in signal phase between each antenna is used to determine the angle. Effectively, the phase difference is retrieved using the TDoA, and the wavelength.

2.2 Geometric Methods

When the measured feature can be directly transformed into spatial information, then it is possible to find the localisation solution using one of the following geometric methods. The explanations in this section assume a 2D map, i.e., the localisation is constrained to the plane XY. Three known positions $[x_1, y_1]^T$, $[x_2, y_2]^T$, and $[x_3, y_3]^T$ are necessary to retrieve an unknown position $[x, y]^T$.

2.2.1 Triangulation

Triangulation is the method of finding the position of a point using two known reference points and the angles of the triangle formed by taking the three points as vertices. The result is retrieved by applying triangle identities from trigonometry.

In a real localisation scenario, as shown in Fig. 2.1, the unknown position $[x, y]^T$ is the mobile device, marked with a circle, and the known positions are the transmitting antennas, marked with squares. The AoA can be measured either at the mobile device or at the transmitting stations if the mobile device can transmit too. For example, the angles $\theta_1$, $\theta_2$ and the known distance $d_{12}$ between the two antennas at $[x_1, y_1]^T$, and $[x_2, y_2]^T$ provide enough information to infer two positions in the plane. The solution ambiguity, marked with a small gray triangle, is solved using $[x_3, y_3]^T$, and either one of the distances $d_{13}$ or $d_{23}$.
2.2.2 Trilateration

Trilateration is the method of finding the position of a point using the geometry of triangles in a similar way as triangulation. While triangulation uses angles and the distance between two reference points, trilateration uses the relative distance of the point to three or more known reference points. These distances are used in a system of equations describing the circles centred at the reference points, the radius of these circles being the known distances to the unknown position.

In a real localisation scenario, as shown in Fig. 2.2, the same set of points used in Fig. 2.1 is again used. The unknown position $[x, y]^T$ is the mobile device, marked with a circle, and the known positions are the transmitting antennas, marked with squares. The distance to each antenna, that is, $r_1$, $r_2$, and $r_3$, can be retrieved either with RSS, ToA or the Round Trip Time. Using for example $r_1$, and $r_2$, the two circles intersect at two positions. The solution ambiguity, marked with a small gray triangle, is solved using $r_3$. The system of quadratic equations describing this example is given by:

\[
\begin{align*}
\sqrt{(x - x_1)^2 + (y - y_1)^2} &= r_1 \\
\sqrt{(x - x_2)^2 + (y - y_2)^2} &= r_2 \\
\sqrt{(x - x_3)^2 + (y - y_3)^2} &= r_3
\end{align*}
\]
If measurements from more than three antennas are available, then some optimisation technique (such as least square estimation) can be used to retrieve the unknown position using all measurements. This is particularly advantageous for noisy measurements and uncertain distance estimation.

2.2.3 Multilateration

Multilateration, also known as hyperbolic positioning, is the method of finding a position using hyperbolae equations, based on the difference of distances. The equations describing circles in the trilateration case are transformed in hyperbolae equations if one is subtracted from another. The intersection of any two hyperbolae defines the point corresponding to the unknown position.

In a real localisation scenario, as shown in Fig. 2.3, the same set of points used in Figs. 2.1 and 2.2 is again used. The unknown position \([x, y]^T\) is the mobile device, marked with a circle, and the known positions are the transmitting antennas, marked with squares.

The difference of distances is retrieved using the TDoA from two antennas and the speed of propagation (usually approximated as \(c\) for air), defining one hyperbola. The third antenna is then used with any of the previous antennas to build a second
Figure 2.3: Multilateration

The equations of hyperbolae usually define two separate curves in $\mathbb{R}^2$. Each hyperbola in 2.3 defines instead only one, since the signal of $\tau_{12}$, and $\tau_{13}$ are already known. This suffices for the solution uniqueness, as can be seen from the hyperbolae with bold black lines in Fig. 2.3. The complementary hyperbolae, which correspond to the signal inversion of the TDoA ($-\tau_{12}$ and $-\tau_{13}$), are also displayed as dashed gray lines. Similarly, the time difference $\tau_{23}$ and $-\tau_{23}$ could be used to find another hyperbolae, displayed with dash-dotted gray lines. The multiple solutions from the triangulation and trilateration cases, marked as gray triangles, also belong to some of
the hyperbolae.

Again, if more than three measurements are available, they can be used to refine the estimation with some optimisation technique. In [32] a method for solving hyperbolic equations in $\mathbb{R}^3$ is given. In that case the equations describe hyperboloid surfaces in 3D space, and a minimum of four known positions are required to retrieve user position. For example, GPS operates in this way.

### 2.3 Pattern Matching

The geometric methods for localisation presented in the last section work well if the measured features can straightforwardly be transduced into spatial information. Their results are often found in a continuous domain.

However, it is common for some localisation systems to discretise the space and to model it at the sampling positions defined by the discretisation. In this way, each position is recorded with some physical quantity associated to it, in the form of a table or list. Such a table is often called a feature map. When the recorded information comes from radio signals, the term radio map is commonly used as well. The set of all recorded positions belongs to the input space, which for a map is usually a Cartesian space, and the set of all stored data belongs to the feature or signal space.

The localisation is made by comparing the actual measurement, i.e., the pattern, with the recorded values in the radio map, often regarded as calibration data. After defining a distance measure (or cost function), a search is made over the radio map in the signal space, looking for the entry on the table that best matches the actual pattern. The corresponding position in Cartesian space recorded with this best match on the radio map is retrieved as a location for this actual measurement. Consequently, the resolution of the sampling positions in Cartesian space is directly related to the best possible resolution for a localisation system. Interpolation may be used to achieve higher accuracy, retrieving positions that need not to be at the radio map positions. However, the accuracy is still subject to the original sampling position density, since the interpolation uses the recorded features/positions as reference. In the literature, pattern matching is also called fingerprinting.

The usual feature recorded in the radio maps is the RSS. However, there are many
possible ways to record this feature. The RSS measurements taken to construct the radio map may be recorded, for example, in the form of a time series, mean values, variances, or statistical distributions.

Throughout this work only the mean values and the statistical distributions are treated. When the feature is recorded in the form of mean values the chosen pattern matching method can be the Nearest Neighbours (NN) or the k-Nearest Neighbours (kNN). If the recorded feature is in the form of probability distributions, then the pattern matching can be done using either Maximum Likelihood (ML) or the Minimum Mean Squared Error (MMSE). The following sections explain each one of them.

2.3.1 Nearest Neighbours

Among all pattern matching methods, the Nearest Neighbours (NN) is one of the simplest. Therefore, it is used for a huge variety of applications such as clusterisation, vector quantisation and self organising maps (SOMs).

The NN algorithm searches in the signal space of the stored patterns for the closest match of an input signal based on a distance measure. The signal space is usually high dimensional and the feature space is usually assumed to be Euclidean. Applied to a localisation scenario the NN algorithm can be defined as:

An area is covered by \( N \) BSs. A radio map is constructed in this area at \( Q \) selected positions \( x_q \), with \( q = \{1, \cdots, Q\} \) and at each of these positions the mean RSS values from \( N \) BSs are calculated and stored in the vector \( \mathbf{p}_q = [p_{q,1}, \cdots, p_{q,N}]^T \). The set \( \{p_{q,1}, \cdots, p_{q,N}\} \) contains the respective RSS values from BS1 to BS\( N \). \( N \) gives the dimension of the signal space.

At some unknown position \( x_M \) a measurement vector \( \mathbf{p}_M = [p_{M,1}, \cdots, p_{M,N}]^T \) is obtained. Then the location \( x \) on the radio map is retrieved using:

\[
x = \arg \min_{x_q} \sum_{n=1}^{N} (p_{q,n} - p_{M,n})^2,
\] (2.5)

which represents the nearest match in feature space to the true position \( x_M \).

The cost function defined at (2.5) can be defined with other distances rather than the Euclidean. The important feature is the minimisation of the defined distance. The estimated position \( x \) belongs to the set defined by \( x_q \).
If the radio map can be described continuously, for example as a set of equations, each one representing a continuous surface for each BS, then the NN search is equivalent to the global minimum search of a functional composed by the difference of these surfaces and their corresponding measurement.

### 2.3.2 k-Nearest Neighbours

The NN algorithm selects one single pattern as winner and delivers it as the best representation among the recorded patterns for a new input feature. Nevertheless, other neighbours can be roughly as far from the input feature as the winner selected by the NN (and in fact, even with multiple identical matches for a minimum distance, only one is retrieved, according to some decision law).

The k-Nearest Neighbours algorithm (kNN) gathers the k best matches to compose a possibly better estimation for the input location than with the usual NN. How these k matches can be combined to deliver one single result depends on the meaning of the information in the input space. For localisation, since the input space represents locations on a map, this combination can be done as the average position of all k matches.

Due to the inherent variation of the measurements, there is no reason to pick only the closest neighbour and reject others with almost the same distance in the signal space [6]. The estimated location for \( x_M \) is calculated as the barycentre of all k positions, thus implicitly acting as a filter to the input noise. The case where \( k = 1 \) reduces to the NN algorithm.

### 2.3.3 Maximum Likelihood

Since the RSS values at \( x_q \) can be modelled as random variables, statistical theory can be used to solve the matching problem. In fact, taking the mean value of the RSS (as for NN and kNN) already assumes random variables, where the focus is placed in one statistical moment only, the mean.

Taking this into consideration, probabilistic localisation deals with the task of determining \( \Pr(x|p_M) \), the probability that a user is at the position \( x \) given the feature \( p_M \), here a RSS measurement. The solution to this problem is found by calculating the
posterior probability over all possible $Q$ locations on the discrete feature map. This is usually accomplished using Bayes rule:

$$
\Pr(x|p_M) = \frac{\Pr(p_M|x) \cdot \Pr(x)}{\sum_{q=1}^{Q} \Pr(p_M|x_q) \cdot \Pr(x_q)}, \quad (2.6)
$$

where the conditional probability $\Pr(p_M|x)$ is retrieved from labelled samples, that is, calibration data retrieved at all $x_q$ locations on the feature map, and used to build the probability density function (pdf) $g(p, x)$ as the recorded feature. $\Pr(x)$ is usually set as a constant, assuming it is uniformly distributed if there is no prior information of where the user can be. The denominator in (2.6) acts as a normaliser.

If the measured feature $p_M$ has dimension $N \neq 1$ and the measurements from $N$ distinct BSs are assumed independent, then $\Pr(p_M|x)$ can be calculated as:

$$
\Pr(p_M|x) = \prod_{n=1}^{N} \Pr(p_{M,n}|x) \quad (2.7)
$$

In this case a unidimensional pdf $g(p_n, x)$ is needed for each BS in order to obtain $\Pr(p_{M,n}|x)$.

A position estimate is retrieved out of these probability densities by searching for the position $x$ where the likelihood $\Pr(p_M|x_q)$ is maximised:

$$
x = \arg \max_{x_q} \Pr(p_M|x_q) \quad (2.8)
$$

This defines the Maximum Likelihood (ML) estimator where, as self explained by its name, the most likely location $x$ is determined based on the observation of the actual measurement $p_M$.

Noteworthy is the symmetry between the ML and the NN method. While the first looks for a maximum in a likelihood function, the latter looks for a minimum of a cost function. In fact, if the measurements at $x_q$ can be assumed to have a Gaussian distribution, the prior probability function $g(p_n|x_q)$ of the RSS measurements for one BS given $x_q$ is given by:

$$
g(p_n|x_q) = \frac{1}{\sigma_{q,n} \sqrt{2\pi}} \exp \left(-\frac{(p_n - \mu_{q,n})^2}{2\sigma_{q,n}^2}\right), \quad (2.9)
$$
where $\mu_{q,n}$ is the mean RSS from BS$_n$ at position $x_q$, and $\sigma_{q,n}$ is the standard deviation at this position. Also, if $\sigma_{q,n} = \sigma_n \forall x_q$, and is constant, then (2.8) reduces to (2.5), as has been shown in [97]

### 2.3.4 Minimum Mean Square Error

Together with ML, the other common way to solve Bayesian estimation problems is using the Minimum Mean Variance Bayesian Estimator or Minimum Mean Square Error (MMSE) Estimator. This method gives an unbiased estimator with minimum mean square error. Considering $x_M$ as the true location and $x$ as the estimated location, the expected value of the mean square error is written as:

$$E[(x_M - x)^2] = \sum_{q=1}^{Q} \|x_q - x\|_2^2 \cdot Pr(x_q|p_M) \quad (2.10)$$

The best estimation for $x_M$, which minimises (2.10) is given by:

$$x = E[x_M|p_M] = \sum_{q=1}^{Q} x_q \cdot Pr(x_q|p_M), \quad (2.11)$$

$E[x_M|p_M]$ being the expected value for $x_M$ given $p_M$. A proof for this is presented in Appendix B.1.

A parallel can be drawn between the kNN and the MMSE methods as was done with the ML and NN methods previously. Both use a composition of the best matches to deliver a final estimate. As to kNN, the number of neighbours $k$ is a user defined parameter. For the MMSE this depends only on the posterior $Pr(x_q|p_M)$, which defines how much weight $x_q$ will have on the final estimate of $x$ as shown in (2.11).

### 2.4 Practical Issues

The most used and known localisation systems, such as the GPS, the RADAR, the LIDAR and the SONAR, require line-of-sight (LOS), that is, a free direct path between the location equipment output and a desired mobile target. The LOS concept may have many interpretations depending on the frequency considered, not always related to
“sight” in the sense of visible light. For example, the SONAR uses acoustic waves, the LIDAR uses LASER beams (either in visible spectrum or infrared), and RADAR and GPS use radio frequency (RF). The necessary condition for LOS is that no material capable of blocking or reflecting a signal from its source is allowed to be between transmitter and receiver, as the time is the feature used in these systems (the same material can be totally transparent or opaque depending on the frequency, like a thick cardboard plate for radio waves or visible light).

2.4.1 Free Field - Line of Sight

The LIDAR, the SONAR, the RADAR and the GPS are suited for open or free field environments using time as feature, as just explained. The measurements in this condition are not affected by the environment where the signal travels. Localisation systems that depend on time measurements must be very precise in order to be accurate. For instance, with electromagnetic waves, considering the speed of light in vacuum, an uncertainty of 10ns on the time measurement represents an uncertainty of 3m in the location estimate. A 1\(\mu\)s measurement uncertainty results in an uncertainty of 300m in location.

None of these systems are used for communication purposes, remaining exclusively for localisation tasks. The communication systems that operate with satellites must have LOS between receivers and the satellite. They usually operate in microwave frequencies and require their directional antennas to be pointed to the transmitter. Archaic communication systems that depended on light required LOS as well (torches between distant observation points on the great wall of China, or light communication between lighthouses and ships).

2.4.2 Non Line of Sight

The requirement of LOS would have restricted a lot the popularity of wireless networks for mobile devices and in fact, most of the wireless communication systems take advantage of Non-Line-of-Sight (NLOS) conditions in order to work. For example, the transmission of short waves can travel around the Earth by bouncing between the Earth and the ionosphere (for other frequencies the ionosphere is transparent and the
radio waves are sent into space).

Electromagnetic waves change their original propagation direction when they reach the boundary between two different media. This effect is known as reflection when the wave is returned into the medium from which it originated, and as refraction when it passes to another medium where the propagation speed is different. Reflection and refraction were first explored with visible light using geometric optics [74], and later extended to other frequencies of electromagnetic waves as a result of Maxwell’s equations [71,72].

Additionally, electromagnetic waves are affected by diffraction, a phenomenon that can be explained by interfering waves and the Huygens-Fresnel principle for wavefronts. This principle states that each point of an advancing wavefront is in fact the centre of a new disturbance and also a new wave source. The advancing wave can be regarded as the sum of all the secondary waves arising from points in the medium already traversed. This explains how electromagnetic waves can “turn” around a corner, and how destructive interference appears causing an effect known as fast fading.

Deflection can also change the direction of electromagnetic waves without a change in the propagation medium (as happens for refraction) in the same way an electron beam is bent inside a cathodic ray tube (CRT). A strong magnetic field, such as those generated by power transmission lines, power transformers and electromechanical devices with rotating fields (such as motors and generators), can cause this phenomenon.

### 2.4.3 Structured Environment - Multi Path

Structured environments differ from free field in the sense that NLOS scenarios are much more frequent than LOS. Large scale outdoor examples include a densely populated city or a campus. Smaller scale indoor examples include buildings with many rooms.

The radio propagation in these environments is strongly influenced by the effects of NLOS, described in the previous subsection. All these effects act together such that one single signal originating on a transmitter reaches a receiver by many different paths as multiple copies of itself. This phenomenon is known as multi path and for many years it was a problem for communication systems, which affected transmission
quality (for analogue TV broadcasting it was common to see “ghosts” on the image). Today, multi path is even used to improve transmission quality by taking advantage of the received information redundancy (if one of the existing communication paths is suddenly lost, there are still other paths to guarantee that the channel is not broken). Examples of some wireless networks that operate under NLOS conditions are WLAN, WiMax, DECT, and GSM.

Multi path strongly affects localisation systems that are based on time measurement. Since the travelled path of a signal is not known, the time information cannot be used directly by the geometric methods described in section 2.2. GPS is one of the systems strongly affected by NLOS scenarios and multi path. In a city with high buildings, it is common to lose sight of all available satellites. Consequently, all the received signals reach the antenna by reflections, refraction and diffraction (considering the particular case of navigation with GPS, if absolutely no signal reaches the antenna, then other information sources must be used in order to estimate the route such as a tachometer and gyroscope).

Localisation systems based on feature maps and pattern matching are less sensitive to multi path, since the information gathered to build the feature map already takes the multi path implicitly into consideration. This makes wireless networks that operate under NLOS conditions an attractive base for localisation systems. Not only is the whole infrastructure already available, but also with these systems the RSS measurement is a standard feature, used in many procedures to establish and assure a communication channel.

In fact, most of the successful indoor localisation systems for sale on the market use the already installed infrastructure of wireless networks and RSS measurements to build a radio map, as for example the Siemens HiPath Location System or the Ekahau Positioning Engine.

Likewise, this work assumes that a wireless communication network is already available and that the feature measurements can be accomplished without extra or proprietary hardware, which impacts directly on implementation costs. It also assumes that the measured feature is RSS for the reasons just explained. Nevertheless, the algorithms proposed here are by no means constrained to this feature only. Other features with spatial gradients, like propagation times or angles of arrival, could be used as well.
2.4.4 Measurement

In the real world any measurement is corrupted by noise. The environment itself can be a source of interference, which will result in noisy measurements. Considering radio measurements, common sources of interference are the proximity to a strong magnetic field source, such as motors, generators, transformers and even power lines (sometimes such interferences are detectable as a 50 or 60Hz peak in the frequency domain). Another noise source, intrinsic to electronic devices, is the thermal noise of semiconductors. Usually the noise is modelled as additive and if its source is well known (for example the 50 or 60Hz from power lines), there can be ways to predict it and thus to cancel it.

Besides the noise, another barrier is imposed by the measurement device. It can happen that the measuring device has a lower sensibility or operates in a smaller range than the range of the actual signal. A practical example is with RSS measurements, where power values weaker than a given threshold are not detected at all, although the signal is actually there. On the implementation side, this absence of measurement can be replaced by a Not-a-Number (NaN), which has logical meaning in many programming languages when a number is otherwise expected. It is also common to saturate the measurement scale on the lowest detectable value, and assign it to all non detected measurements.

Often noise is modelled as Gaussian, which implies that the measurement itself is Gaussian distributed. Though for some cases this may be a reasonable approximation, for indoor localisation this assumption must be treated carefully. A straightforward example is if the saturation is applied at the lowest detectable value. Even if the noise is indeed Gaussian, the non-linearity imposed by the saturation will distort and mask the measurements. A Gaussian model based on these distorted measurements will not correspond to reality.

Therefore, some suitable test should be applied before making assumptions that may not correspond to reality. And for that, there are many ways to check if the measurements do belong to a Gaussian distribution or not: from statistics, the Shapiro-Wilk test checks the null hypothesis that a set of samples originates from a normally distributed population [92,93]. Also from statistical properties of the Gaussian distri-
bution, a check of the kurtosis and the skewness values can be used to test a set of samples (for Gaussian distributed both the excess kurtosis and skewness are equal to 0). The D’Agostino-Pearson $K^2$ normality test already encapsulate both skewness and kurtosis to check a null hypothesis of normality over a set of observations [31].

Even assuming that ideally all noise could be eliminated, there are still some considerations that affect measurements in indoor environments, which have been regarded as static up to now. Due to the movement of people inside a building, which has as a consequence the constant closing and opening of doors and windows, and the displacement of lifts (and even cars passing by outside or even inside the building if it has a garage floor), an indoor environment is anything but static.

Just one single door can already determine two distinct states where a signal path may take two different ways between a static transmitter and a static receiver, thus with two different measurement possibilities, one for each state. A corridor with all doors closed can act like a waveguide, while if one or more doors are open, the signal can escape and this waveguide effect would be lost. Considering each door as a “two state device” (disregarding, for simplicity, that this door aperture is a continuous variable and not discrete), it is easy to understand the dimension of this problem considering all possible states with door closed/opened. In addition, even a person can define two detectable states, since the human body can strongly affect radio transmissions, if an office room is occupied by one person or none.

So far, all localisation systems assume that the environment is static. Once the radio map is built, it is assumed valid from then on. This assumption is obviously naive and some systems need to be recalibrated after some time. In fact, an indoor environment can change from a static state to another by displacements of furniture and division walls. Such a change would ideally require a recalibration procedure, which makes localisation systems based on radio maps very costly to be kept accurate over long periods of operation.

In this thesis, such environmental changes are taken into consideration, and in fact used to adapt the radio map without the need for recalibration. To the knowledge of the author, this is the first work to dynamically treat this case.
Chapter 3

SLL: Simultaneous Localisation and Learning

3.1 Learning Systems

The accuracy of a localisation system depends on the radio map resolution and on the quality of the recorded information. Even using an existing infrastructure, the cost imposed by the calibration phase, which must be performed before the system start, is still a significant barrier to the deployment of indoor localisation systems.

The challenge with reducing the cost of such a localisation system can be understood by the following questions. How to build a feature map with the least amount of calibration and still achieve a reasonable accuracy? Could the calibration phase be shortened (or even eliminated) so that the localisation system could be used right after its start?

The answer to these questions relies on learning. The system should be able to learn the feature map in a consistent and intelligent way so that the calibration effort can be eliminated. The proposed way of learning is to use Self Organising Maps (SOMs), feeding it with unlabelled samples as input patterns.

The new algorithms introduced here, i.e., SLL and SPLL, work according to the following scenario. A coarse feature map is constructed based on little information
and rough modelling prior to system start. A measurement is taken and then used to locate a user, based on this coarse initial feature map. The found location is then used as centre for a learning step, where the neighbourhood surrounding this centre is adapted towards the measurement. These operations continue repeatedly at each new measurement, thereby improving the feature map.

In the following sections a brief overview of SOMs is given. Then the SLL is presented as a solution to the self-calibration task, together with its algebraic properties and the statistical conditions required for its successful use. Finally, the SPLL is introduced as the evolution of SLL.

### 3.1.1 Introduction to SOM

SOMs are a special class of neural networks, which are based on competitive learning. In a SOM, the neurons are placed at the nodes of a lattice that has usually one or two dimensions (1D or 2D), known also as latent space. The neurons become selectively adapted to various input patterns in the course of an unsupervised competitive learning process. The locations of the winning neurons in latent space become ordered with respect to each other in such a way that a meaningful coordinate system for different input patterns is created over the lattice [59]. A SOM is therefore characterised by the formation of a topological map where the input patterns are mapped from the input space into the latent space, preserving the intrinsic statistical features contained in the input patterns [51].

The principal feature of Kohonen’s SOM is the automatic and adaptive mapping of signals from the input space into the latent space, using the input signals to perform a parametric regression over the neurons, fitting them to the distribution associated to the input samples. This mapping forms clusters in the latent space, which preserve the topological relations of the data in the input space. This is called self-organisation.

Learning is achieved by performing iteratively three steps in addition to the initialisation: competition, cooperation and adaptation. During the initialisation the synaptic weights in the neural network are randomly set, if no other initialisation is specified.

In the competitive step a winning neuron $c$ with the weight vector $m_c = [m_{c1}, \ldots, m_{cn}]$ in the $n$ dimensional input space is selected such that it has the smallest cost with
respect to a given input feature vector $\xi = [\xi_1, \cdots, \xi_n]^T$. The cost is usually calculated using some distance measure in input space between $\xi$ and all weights $m_i$, that is, $c = \arg \min \{d(\xi - m_i)\}$, with $d$ as the distance measure (as the Euclidean distance for most practical applications) and the index $i$ going through all neurons in the lattice. The winning neuron $c$ will be the centre for the adaptation process.

The cooperation determines which neurons will be adapted together with the winning neuron $c$. A neighbourhood function $h_{ci}(k)$, dependent on the discrete time step $k$, is used to find the neuron $i$ close to the winner $c$ and to weigh it accordingly with the distance to the winner in the lattice. A typical choice for the neighbourhood function in 1D problems is the constant function, set to a constant $\alpha(k)$ for the winner and for an equal number of neighbours, forward and backward (usually just 2 neighbours are taken). For 2D or 3D maps a Gaussian function is usually chosen, such that:

$$h_{ci}(k) = \alpha(k) \cdot \exp \left( -\left( \frac{d_{ci}}{2 \cdot \sigma(k)} \right)^2 \right), \quad (3.1)$$

where $\alpha(k)$ is the learning rate, $\sigma(k)$ is the effective width of the topological neighbourhood, both dependent on $k$. $d_{ci}$ is the distance in latent space from neuron $i$ to neuron $c$ at the centre. The adaptation law, given by

$$m_i(k+1) = m_i(k) + h_{ci}(k) \cdot (\xi(k) - m_i(k)), \quad (3.2)$$

ensures that the response of the winning neuron to the subsequent application of a similar input pattern is enhanced [51].

The adaptive process consists of two phases: the self-organising or ordering phase and the convergence phase. In the ordering phase the topological ordering of the weight vectors takes place. During this phase the learning rate and the neighbourhood area should decrease. The neighbourhood area goes from complete coverage to a few neurons or even to the winning neuron itself. In the convergence phase the fine tuning of the feature map takes place in order to provide an accurate statistical quantification of the input space. The learning rate should stay constant or it could decay exponentially [51].

The Kohonen algorithm is surprisingly resistant to a complete mathematical study (cf. [29]). The only thorough analyses could be achieved for a 1D input space and a
1D neuron lattice. For higher dimensions, the results are only partial.

3.2 SLL: Main Algorithm

The Simultaneous Localisation and Learning (SLL) was designed to use the mean value of RSS as recorded feature. Hence the chosen localisation method was the NN search, as explained in section 2.3.1, and an explanation of the algorithm is as follows.

3.2.1 The Initialisation

At the beginning the feature map must be populated so that the first location query can be performed. While for common localisation systems this is done during the calibration phase, for the learning systems here presented this is achieved with an artificial model.

The initial model plays a major role for the learning systems here presented. Since it replaces the calibration phase, it should at least correspond in some extent to reality, so that the localisation accuracy is rough but still acceptable. A poorly chosen model, for example one with a constant feature value through all space, can cause the algorithm to fail.

To construct the model, it suffices to have information about the size of the map, the position of the BSs and the positions of the \( Q \) sampling points if the feature map is discrete. The model must represent physically plausible conditions for radio propagation, for example with the highest RSS value placed at the BS or at the nearest feature map position to a BS (for the case where the BS position itself does not belong to the feature map).

If these conditions are fulfilled, then the learning can proceed. It makes no sense to invest too much time creating a refined model, since the learning will perform this refinement by itself. This will be later verified by experiments with the following two models in this chapter. They are both simple but differ in the amount of information necessary to be built.
3.2.1.1 Radial Model

The Radial Model (RM) is one of the simplest that can be implemented. It imposes a radial symmetry around each BS, and a monotonic decay of the field strength with increasing distance. The RSS from BS $n$ as a function of the relative distance $d_n$ to any position can be written as:

$$p_n = p_{\text{out},n} - D(d_n, \gamma_n),$$  \hspace{1cm} (3.3)

where $p_{\text{out},n}$ is the output power of BS $n$ and $D(d_n, \gamma_n)$ is a monotonic function, which usually describes a linear or logarithmic decay, as for example in (2.1). $\gamma_n$ is a parameter that controls how steep the decay of the RSS field is.

Once a proper function $D$ is chosen, the only information required are the BS positions, and their output power $p_{\text{out}}$. Optionally the parameters of $D$ could be fitted to any measurement information that may be available at this point, as for example mutual measurements from BSs.

An illustrative example of RM is displayed in Figure 3.1 considering two BSs, both with $p_{\text{out}} = -20\text{dBm}$ for two different forms of the function $D$: in Figure 3.1a $D$ has a linear profile, while in Figure 3.1b a logarithmic profile is shown. In both cases the radial symmetry around each BS, which named this model, can be observed.

In [68] the RM is referred to as Linear Model (LM), since the logarithmic decay modelled by (2.1) is linear with a logarithmic scale for the distance, and some calibra-
tion data was in fact fitted to a line. In order to generalise the definition of this model when \( D \) is not necessarily linear, the term Radial Model is preferred as in [16].

3.2.1.2 Dominant Path Model

The initial model can be made more accurate when more information is introduced into it. If map information is available, that is, not only a background figure with representations of spatial division imposed by walls, rooms and corridors, but semantic information specifying exactly where these divisions are and how they affect radio propagation, then this information can be integrated into the model.

The RM considers a direct path between the transmitter and a receiver position. Through the map information, it is possible to know how many walls this direct ray crossed until reaching the receiver. With information about how the walls attenuate the radio signal, a corrected value for the RSS can be calculated. This model, called as Multi Wall Model (MWM), usually leads to a RSS underestimation in many cases [68].

Ray tracing algorithms take into consideration every possible path between transmitter and receiver, requiring a high computational effort [37, 38, 42, 96, 103, 104].

In Figure 3.2, the BS emits many possible rays to the receiver Rx. The direct path penetrates three walls and some other paths bounce, reflect and refract to reach Rx, each of them with a single contribution to the RSS that will be added by ray tracing. The RSS estimation is indeed more accurate, but ray tracing programs need not only precise geometric information, but also information about the materials in the environment and their properties that affect radio waves (such as conductivity, refraction coefficient, permeability, permittivity). This information is used to compute the paths taken by waves leaving the transmitting antenna in all directions, which is very computationally intensive and time consuming. One path particularly has the strongest contribution among all others, presenting the optimal combination of distance, number of walls crossed and other attenuation factors, being called the dominant path.

The Dominant Path Model (DPM) [68, 102] aims to find the dominant path, which could be a direct path as well, and use this single contribution to estimate the RSS. This overcomes the underestimation of only using the direct path and is considerably faster than ray tracing, since the information load to compute the dominant path is considerably smaller. In [97, 102] the algorithm to find the dominant path is given and
Figure 3.2: Possible paths between a transmitter and a receiver will not be described here.

3.2.2 The Learning Phase

Once the initial model is available the learning phase can begin. The learning is an iterative process, and it is natural to define it as a function in discrete time. In this way $p_k(x)$ describes the RSS propagation through space at the discrete time $k$, and as a function of $x$, a position in some fixed reference frame. The dimension of $p_k(x)$ defines the number of BSs considered for the localisation scenario such that:

$$p_k(x) = [p_{1,k}(x), p_{2,k}(x), \cdots, p_{N,k}(x)]^T$$ (3.4)

Then $p_0(x)$ represents the model at the time $k = 0$, that is, the initial model. The measurement $p_M$ is associated with $x_M$, a (not necessarily known) measurement position. $p_{M,k}$ is a measurement taken at the discrete time $k$.

Starting from (3.4), the SLL is defined by the following feedback or update law:

$$p_{k+1}(x) = p_k(x) + f_{c,k+1} \cdot (p_{M,k+1} - p_k(x)),$$ (3.5)

where $f_{c,k} = f(x_{c,k}, x, \kappa, \phi)$ is a function of the centring position $x_{c,k}$ at time $k$, of $x$, and of the SLL control variables $\kappa$ and $\phi$. $f_{c,k}$ spatially bounds and weights the update based on the difference between the actual measurement $p_{M,k+1}$ and the present model $p_k(x)$.

$f_{c,k}$ can have different forms, given for example, by a line, a polynomial or Gaussian bell. Its important characteristic is that it must be symmetric around $x_{c,k}$ and must have its magnitude bounded by the interval $[0; \kappa]$, with $\kappa \leq 1$. If the distance from $x$
to $x_{c,k}$ is greater than $\phi$ then $f_{c,k} = 0$. The function $f_{c,k}$ reaches its maximum at $x_{c,k}$ with value $\kappa$, and falls to smaller values until the boundary defined by $\phi$. The location $x_{c,k} = x_{c,k}(p_{M,k})$ corresponds to the measurement $p_{M,k}$. The determination of $x_{c,k}$ depends on the localisation technique chosen as introduced in chapter 2. Figure 3.3 shows two qualitative examples of the function $f_{c,k}$ and the meaning of its arguments in a 1D and 2D context.

The performance of the SLL algorithm in (3.5) is influenced by the initial model $p_{0}(x)$, the quality of the measurement $p_{M}$, its own parameters $\kappa$ and $\phi$, together with the localisation approach, which will place $f_{c}$ by determining $x_{c}$.

A deeper analytical study of the algorithm is required in order to better understand how these aspects influence the SLL, so that strategies to guarantee the convergence of the algorithm can be developed. Here the term convergence is used in a wide sense, i.e., the final range of the learned radio map is tightly bounded.

In the following sections some boundary limits will be provided algebraically, considering the initial model and the quality of measurement. Laboratory simulations reveal that, when the localisation approach is no longer perfect, the sequence of measurements has to obey certain statistical properties in order to make the final radio map fall together with the real world measurements. These statistical properties are then used to define rules for the real world application of SLL.
3.2.2.1 Algebraic properties

The SLL can be written in a closed form that can be derived from its recursive formulation in (3.5), as the next theorem states.

**Theorem 3.1.** The recursive SLL formulation in (3.5) has the following closed form:

$$p_k(x) = p_0(x) + \sum_{j=1}^{k} \left( f_{c,j} \cdot (p_{M,j} - p_0(x)) \prod_{i=j+1}^{k} (1 - f_{c,i}) \right), \quad (3.6)$$

with

$$\prod_{i=a}^{b} (\cdot) = 1, \forall a > b, \quad (3.7)$$

The proof to this theorem is made by induction:

**Proof.** The first and second iterations can be directly written from (3.5) as:

$$p_1(x) = p_0(x) + f_{c,1} \cdot (p_{M,1} - p_0(x))$$
$$p_2(x) = p_0(x) + f_{c,1} \cdot (p_{M,1} - p_0(x)) (1 - f_{c,2}) + f_{c,2} \cdot (p_{M,2} - p_0(x)),$$

which conforms with the closed form given in (3.6). Assuming that the closed form of the update law (3.6) at step $k$ is true, the step $k + 1$ is computed using the recursive formula from (3.5):

$$p_{k+1}(x) = p_0(x) + \sum_{j=1}^{k} \left( f_{c,j} \cdot (p_{M,j} - p_0(x)) \prod_{i=j+1}^{k} (1 - f_{c,i}) \right)$$
$$+ f_{c,k+1} \cdot \left( p_{M,k+1} - p_0(x) - \sum_{j=1}^{k} \left( f_{c,j} \cdot (p_{M,j} - p_0(x)) \prod_{i=j+1}^{k} (1 - f_{c,i}) \right) \right)$$

Reordering, combining the sum-terms and placing the term $(1 - f_{c,k+1})$ into the summation gives:

$$p_{k+1}(x) = p_0(x) + f_{c,k+1} \cdot (p_{M,k+1} - p_0(x))$$
$$+ \sum_{j=1}^{k} \left( (1 - f_{c,k+1}) f_{c,j} \cdot (p_{M,j} - p_0(x)) \prod_{i=j+1}^{k} (1 - f_{c,i}) \right)$$

31
Moving the term \((1 - f_{c,k+1})\) into the product-term gives:

\[
p_{k+1}(x) = p_0(x) + f_{c,k+1} \cdot (p_{M,k+1} - p_0(x)) + \sum_{j=1}^{k} \left( f_{c,j} \cdot (p_{M,j} - p_0(x)) \prod_{i=j+1}^{k+1} (1 - f_{c,i}) \right)
\]

With the property (3.7) the update law at step \(k + 1\) is achieved in the same form as the update law at step \(k\); which concludes the induction.

\[
p_{k+1}(x) = p_0(x) + \sum_{j=1}^{k+1} \left( f_{c,j} \cdot (p_{M,j} - p_0(x)) \prod_{i=j+1}^{k+1} (1 - f_{c,i}) \right)
\]

Defining the utility functions:

\[
F_k = \sum_{j=1}^{k} \left( f_{c,j} \prod_{i=j+1}^{k} (1 - f_{c,i}) \right), \quad (3.8)
\]

and

\[
P_{M,k} = \sum_{j=1}^{k} \left( f_{c,j} \cdot p_{M,j} \prod_{i=j+1}^{k} (1 - f_{c,i}) \right). \quad (3.9)
\]

Then (3.6) can be compactly written as:

\[
p_k(x) = p_0(x) - p_0(x)F_k + P_{M,k}. \quad (3.10)
\]

**Limit Value**  Given the formulation (3.10) of the SLL, it is possible to understand what happens to the feature map, and how it relates to the initial model, represented by \(p_0(x)\), when the time \(k\) advances, particularly when \(k \to \infty\).

For that, the recursive formulation of (3.8), given by the next lemma, is useful:

**Lemma 3.1.** Considering the utility function \(F_k\) in closed form as in (3.8), then its recursive formulation is written as:

\[
F_{k+1} = F_k \cdot (1 - f_{c,k+1}) + f_{c,k+1} \quad (3.11)
\]

**Proof.** Assuming that the sequence starts from 0, then the first terms of (3.8) are
written as:

\[
\begin{align*}
F_0 &= 0 \\
F_1 &= f_{c,1} \\
F_2 &= f_{c,1} \cdot (1 - f_{c,2}) + f_{c,2} = F_1 \cdot (1 - f_{c,2}) + f_{c,2}
\end{align*}
\]

which corresponds to the recursive formulation.

Likewise as in Theorem 3.1, assuming that the recursive formulation is correct, then the term at \(k + 1\) is calculated applying the closed formula at \(k\) into the recursive formula:

\[
F_{k+1} = \sum_{j=1}^{k} \left( f_{c,j} \prod_{i=j+1}^{k} (1 - f_{c,i}) \right) \cdot (1 - f_{c,k+1}) + f_{c,k+1}
\]

Placing the term \((1 - f_{c,k+1})\) into the summation gives:

\[
F_{k+1} = \sum_{j=1}^{k} \left( (1 - f_{c,k+1}) \cdot f_{c,j} \prod_{i=j+1}^{k} (1 - f_{c,i}) \right) + f_{c,k+1}
\]

And gathering it to the product follows:

\[
F_{k+1} = \sum_{j=1}^{k} \left( f_{c,j} \prod_{i=j+1}^{k+1} (1 - f_{c,i}) \right) + f_{c,k+1}
\]

And finally placing \(f_{c,k+1}\) into the sum:

\[
F_{k+1} = \sum_{j=1}^{k+1} \left( f_{c,j} \prod_{i=j+1}^{k+1} (1 - f_{c,i}) \right),
\]

which corresponds to the closed form in (3.8) applied for the time \(k + 1\).

This result will be used to prove the next theorem:

**Theorem 3.2.** When \(k \to \infty\) then \(\lim_{k \to \infty} F_k = 1\)

*Proof.* The recursive form for \(F_k\) in (3.11) is the start of this proof. The evolution of \(F_k\) can be analysed, without loss of generality, at one particular fixed point, for example
Making $f_{c,k+1}(\mathbf{x}_t) = \alpha_{k+1}$, where $\alpha_{k+1} \leq \kappa < 1$, then (3.11) turns into the equation of an exponential filter:

$$F_{k+1}(\mathbf{x}_t) = F_k(\mathbf{x}_t) \cdot (1 - \alpha_{k+1}) + \alpha_{k+1}$$

(3.12)

In general, an exponential filter can be written as:

$$y_{k+1} = y_k \cdot (1 - a) + a \cdot u_k,$$

(3.13)

where $y_k$ is the filter output, $a$ is the filter parameter and $u_k$ the filter input signal. If $0 < a < 1$, and $u_k = u$, constant $\forall k$, then $y_k$ reaches $u_k$ asymptotically and exponentially. A proof for this is presented at Appendix B.2.

Comparing (3.13) with (3.12) it can be stated that (3.12) is an exponential filter with a time varying parameter and with input signal $u_k = 1 \forall k$. This means that (3.12) reaches 1 asymptotically. The time varying parameter $\alpha_{k+1}$ only affects the speed with which the filter reaches 1 and it is directly dependent on the sequence of selected $f_{c,k+1}$ and thus their values at $\mathbf{x}_t$.

This result can be extended from $\mathbf{x}_t$ to all points in the interval where the $f_c$s are defined, usually all points inside the boundaries imposed by the feature map. The function $F_k$ will then reach 1 for all points belonging to this interval when $k \rightarrow \infty$.

In [13,14] it has been proven that $\lim_{k \rightarrow \infty} F_k = 1$ inside a closed interval belonging to $\mathbb{R}$. This result is extended to higher dimensions by Theorem 3.2, as long as the weighting function $f_{c,k}$ has the properties explained at the beginning of this section and that the series of $\mathbf{x}_{c,k}$s, where the $f_{c,k}$s are centred, cover repeatedly the entire closed interval.

Figure 3.4 exemplifies qualitatively the result of Theorem 3.2 for an interval defined in $\mathbb{R}$ and in $\mathbb{R}^2$. In Figure 3.4a a random sequence of $f_{c,k}$s with a shape as presented in Figure 3.3a ($\kappa = 0.1$) makes $F_k$ in the whole interval $[0; L]$ converge to 1 with growing $k$. In Figure 3.4b the shape of the $f_{c,k}$s are as presented in Figure 3.3b with $\kappa = 0.1$ and 3 stages of $F_k$ can be observed: at the very beginning, where the surface defined at $[0; L_x] \times [0; L_y]$ is fully covered with $f_c$s; at a middle stage, halfway to the convergence; and finally when the whole surface interval converges to 1.
Figure 3.4: Convergence of $F_k$ to 1 (a) with $F_k \in \mathbb{R}$ and (b) $F_k \in \mathbb{R}^2$

$P_{M,k}$, in contrast to $F_k$, cannot reach a steady state. Each measurement $p_{M,j}$, as it appears in (3.9), pushes $P_{M,k}$ towards $p_{M,j}$ inside the hypersphere centred in $x_{c,j}$ and with radius $\phi$. Since the measurements vary through space, $P_{M,k}$ continuously changes.

In this way, when $k \to \infty$, $p_k(x)$, as given in (3.10) tends to:

$$\lim_{k \to \infty} p_k(x) = p_0(x) - p_0(x) + \lim_{k \to \infty} P_{M,k} = \lim_{k \to \infty} P_{M,k}$$

(3.14)

This shows an important result of SLL: eventually, the initial model $p_0(x)$ will be replaced entirely by $P_{M,k}$, a term that depends on the measurements and on the location estimation. Since $p_0(x)$ disappears with increasing iterations, there is no need to make the initial model extremely precise. It suffices to start with a coarse and hence relatively simple model, but with a reasonable location accuracy, as explained in subsection 3.2.1.

Another effect of SLL is that old measurements have a smaller contribution to $P_{M,k}$ than newer ones. This can clearly be seen in (3.9), where the $p_{M,j}$s are multiplied by products of $(1 - f_{c,i}) \leq 1 \forall i$. The older the measurements are, the greater is the number of terms in the product, which will tend to zero. The consequence is that the model is always updated, as long as new measurements are considered and as the control parameters are non-zero.
**Measurement noise** Assuming that each measurement $p_M$ is corrupted by stationary Gaussian noise $\zeta(x) = [\zeta_1(x), \ldots, \zeta_n(x)]^T$ with mean $\mu(x) = [\mu_1(x), \ldots, \mu_n(x)]^T$ and variance $\sigma^2(x) = [\sigma_1^2(x), \ldots, \sigma_n^2(x)]^T$, it is desirable to know the remaining effect of this noise after some iterations of the SLL.

Returning to (3.5), the recursive equation regarding the noise $\zeta_{k+1} = \zeta(x_{M,k+1})$ at the new measurement position $x_{M,k+1}$ becomes:

$$p_{k+1}(x) = p_k(x) + f_{c,k+1} \cdot (p_{M,k+1} + \zeta_{k+1} - p_k(x)),$$

(3.15)

which, similarly to (3.6), leads to the closed form:

$$p_k(x) = p_0(x) + \sum_{j=1}^{k} \left( f_{c,j} \cdot (p_{M,j} + \zeta_j - p_0(x)) \prod_{i=j+1}^{k} (1 - f_{c,i}) \right)$$

(3.16)

The noise term can be separated from (3.16) defining the utility function

$$Z(x, x_{M,1:k}, x_{c,1:k}) = Z_k = \sum_{j=1}^{k} \left( f_{c,j} \cdot \zeta_j \prod_{i=j+1}^{k} (1 - f_{c,i}) \right),$$

(3.17)

such that the following short form is attained using (3.8) and (3.9):

$$p_k(x) = p_0(x) - p_0(x)F_k + P_{M,k} + Z_k,$$

(3.18)

which corresponds to (3.10) with the extra term $Z_k$ modelling the influence of the measurement noise. It is important to note that $Z_k$ depends not only on the considered location $x$, but also on the sequence of true measurement locations, defined by $x_{M,1:k} = \{x_{M,1}, \ldots, x_{M,k}\}$ as well as on the sequence of estimated locations, defined by $x_{c,1:k} = \{x_{c,1}, \ldots, x_{c,k}\}$, where the weighting functions are centred.

The similarity between $P_{M,k}$ in (3.9) and $Z_k$ in (3.17) is notable. They differ only in the scalar term introduced with new iterations: i.e., $p_{M,j}$ and $\zeta_j$, respectively.

$Z_k$ cannot reach a steady state for the same reason as $P_{M,k}$, i.e., since $\zeta_j$ changes, the resulting exponential filter never converges. However, departing from the assumption that each $\zeta_j$ is an independent Gaussian random variable, it is possible to calculate expectations of mean and variance of $Z_k$ based on the mean $\mu_j$ and variance $\sigma_j^2$ of each
ζ_j.

For one particular fixed point x_t, (3.17) shows that Z(x_t, x_M,1:k, x_c,1:k) = Z_k(x_t) is formed as a weighted sum of random variables. And therefore, the following properties for linear operations on independent random variables can be used, provided that a and b are scalars:

\[
\begin{align*}
\text{mean}\{a + b\zeta_i\} &= a + b\mu_i \\
\text{var}\{a + b\zeta_i\} &= b^2\sigma_i^2 \\
\text{mean}\{\zeta_i + \zeta_j\} &= \mu_i + \mu_j \\
\text{var}\{\zeta_i + \zeta_j\} &= \sigma_i^2 + \sigma_j^2
\end{align*}
\]

In this way, using the recursive formulation for (3.17):

\[
Z_{k+1}(x_t) = Z_k(x_t) \cdot (1 - f_{c,k+1}) + f_{c,k+1} \cdot \zeta_{k+1},
\]

setting \(\text{mean}\{Z_k(x_t)\} = M_Z(x_t, x_{M,1:k}, x_{c,1:k}) = M_{Z,k}(x_t)\) and \(\text{var}\{Z_k(x_t)\} = S^2_{Z,k}(x_t)\), using the properties above listed, and assuming that the noise has same magnitude inside of a radius \(\phi\) centred on \(x_t\), it is possible to express the mean of \(Z_k(x_t)\) recursively as:

\[
M_{Z,k+1}(x_t) = M_{Z,k}(x_t) \cdot (1 - f_{c,k+1}(x_t)) + f_{c,k+1}(x_t) \cdot \mu(x_{M,k+1}),
\]

and similarly for its variance as:

\[
S^2_{Z,k+1}(x_t) = S^2_{Z,k}(x_t) \cdot (1 - f_{c,k+1}(x_t))^2 + f_{c,k+1}(x_t) \cdot \sigma^2(x_{M,k+1})
\]

Comparing (3.20) with the recursive formulation for \(F_k\) in (3.12), it follows that (3.20) also has the form of an exponential filter with the variable parameter \(f_{c,k+1}(x_t)\) and with the variable input \(\mu(x_{M,k+1})\). \(\lim_{k \to \infty} M_{Z,k}(x_t) = \mu\) holds for constant \(\mu(x_{M,k}) = \mu\ \forall k\), constant measurement location \(x_M\), and constant estimated location \(x_c\).

Notwithstanding the similarity between (3.20) and (3.21), the latter cannot be treated as an exponential filter due to its quadratic terms. Even if \(\sigma^2\) is constant in all space, \(S^2_{Z,k}(x)\) will vary according to the sequence of \(x_c\)’s. However, \(S^2_{Z,k}(x)\) is upper-bounded by a maximum value. This maximum can be estimated considering a constant update centre, i.e., \(x_{c,k} = x_c\) for all \(k\) and assuming space-invariant and therefore also time-constant noise, i.e., \(\sigma^2(x) = \sigma^2\) and \(\mu(x) = \mu\).
Since $x_{c,k}$ is constant in time, so too is $f_{c,k} = f_c$ for all $k$ (assuming that neither $\kappa$ nor $\phi$ vary with time). The recursive equation for $S^2_{Z,k+1}$ can be written as:

$$S^2_{Z,k+1} = S^2_{Z,k} \cdot (1 - f_c)^2 + f_c^2 \cdot \sigma^2$$  \hspace{1cm} (3.22)

Assuming a steady state, i.e., $S^2_{Z,k+1} = S^2_{Z,k} = S^2_{Z,\text{steady}}$:

$$S^2_{Z,\text{steady}} = \frac{f_c^2 \cdot \sigma^2}{1 - (1 - f_c)^2} = \frac{f_c \cdot \sigma^2}{2 - f_c},$$  \hspace{1cm} (3.23)

holds. In particular at the position $x = x_f = x_c$:

$$S^2_{Z,\text{steady}}(x_c) = \frac{\kappa \cdot \sigma^2}{2 - \kappa},$$  \hspace{1cm} (3.24)

which is the maximum for this function.

Considering that $\kappa \in [0; 1]$, and that the upper bound is given by (3.24), it is easy to verify that $S^2_{Z,\text{steady}} \leq \sigma^2$. This indicates that the variance of $Z_k$ at one particular position will be at most $\sigma^2$, and that only if $\kappa = 1$.

The important result is the noise reduction property of SLL: by exponential filtering and spatial weighting due to $f_c$, the noise variance of the learned feature map is reduced in comparison with the measurement noise. At one particular position $x_f$, this noise averaging is achieved not only using the single measurements at $x_f$, but also using the noisy measurement of neighbouring positions, with update centres $x_c$s located at distances from $x_f$ smaller than $\phi$.

**Limit Area for Perfect Localisation** The concept of limit area appears only if one dimension in space is considered and with perfect localisation. The extension to more dimensions would result in limit hyper volumes and will be discussed on the end of this section.

Considering only one BS, a perfect localisation implies that, for a given measurement $p_{M,k}$, the associated position $x_{c,k}$ corresponds exactly to the real measurement position $x_{M,k}$, i.e., $x_{c,k} = x_{M,k}$. The measurements follow a propagation law $B$:

$$p_M(x) = B(x, p_{\text{out}}, \gamma),$$  \hspace{1cm} (3.25)
where $p_{\text{out}}$ is the output power of the BS and $\gamma$ is some attenuation factor. The signal propagation is assumed to be monotonic, i.e., $\partial B/\partial d_{\text{BS}}$ is always negative, with $d_{\text{BS}}$ being the distance from a position $x$ to the BS. Without loss of generality, the BS is assumed to be placed on the left side of the feature map such that the measurement at $x_t - \phi$ is:

$$p_M(x_t - \phi) = p_{M+} = B(x_t - \phi, p_{\text{out}}, \gamma),$$

(3.26)

$x_t$ being a fixed position and the measurement at $x_t + \phi > x_t - \phi$ is:

$$p_M(x_t + \phi) = p_{M-} = B(x_t + \phi, p_{\text{out}}, \gamma) < p_{M+},$$

(3.27)

and therefore $p_{M+} > p_{M-}$.

**Perfect initialisation** A perfect initialisation means that the starting feature map at the instant $k = 0$ has exactly the same propagation profile as the measurements. Hence, the start model at some position $x_t$ is:

$$p_0(x_t) = B(x_t, p_{\text{out}}, \gamma)$$

(3.28)

As the weighting function $f_c$ has a bounded support, the measurement positions $x_M$ which can change the radio map at the considered position $x_t$ belong to the interval $[x_t - \phi; x_t + \phi]$. Hence, there are two measurement cases to consider:

1. If $x_M$ lies outside of the support of the weighting function $f_c$, i.e., $x_M \notin [x_t - \phi; x_t + \phi]$, then $f_{c,k+1}(x_t) = 0$ and the update law in (3.5) results in $p_{k+1}(x_t) = p_k(x_t)$. In this case no update is made at $x_t$.

2. If $x_M \in ]x_t - \phi; x_t + \phi[$ then $f_{c,k+1}(x_t) \in ]0; \kappa]$ and the update will cover $x_t$. Accordingly, the update law reduces the difference between the feature map at $x_t$ and the current measurement: $|p_{k+1}(x_t) - p_{M,k+1}| \leq |p_k(x_t) - p_{M,k+1}|$. Due to perfect initialisation, for $k = 0$ the model will stay unchanged at $x_t$ only if the measurement equals the radio map, i.e., $p_M = p_0(x_t)$ and $x_M = x_t$. In this case no improvement can be accomplished since the radio map is already perfect. For all other measurement values in the interval $[p_{M+}; p_{M-}]$ the model will be disturbed
Figure 3.5: Limit area for a linear (a) and logarithmic model (b)

at $x_f$ towards the actual measurement. Since $p_{M,k+1} \in [p_{M^-}; p_{M^+}]$ and considering
the assumed monotony, $p_{k+1}(x_f)$ is also limited by this interval. The maximum
positive disturbance at $x_f$ is given by $p_{M^+}$ and the maximum negative disturbance
is given by $p_{M^-}$. $p_{M^+}$ or $p_{M^-}$ can be obtained, if and only if measurements are
repeatedly taken either at $x_f - \phi$ or at $x_f + \phi$, respectively.

Since there is no assumption on the placement of $x_f$, this result can be generalised
to the whole feature map interval $[0; L]$. Thus an upper and lower limit curve can be
defined for the whole interval $[0; L]$, where $p_k(x)$ is not expected to cross. The feature
map is restricted to a limit area, which is defined by moving the true propagation
model by $\phi$ to the left and to the right:

$$B(x + \phi) \leq p_k(x) \leq B(x - \phi)$$

Figure 3.5 shows a qualitative example of how the limit area is achieved for two
distinct measurement profiles: in Figure 3.5a the profile is linear, while in Figure 3.5b
the profile is logarithmic. The initialisation is perfect, so the initial model is coincident
with the measurement profile. In both plots the flattening effect due to the repeated
updates at the same place are observed for distinct iteration steps $k$. Despite the local
disturbance imposed by this flattening, the actual feature map never crosses the limit
area boundaries, plotted as the dashed dotted lines.
If the measurements are not taken repeatedly at the same position, i.e., any position inside the interval $[0; L]$ could be taken, then the radio map $p_k(x)$ still stays inside the defined limit area.

An illustrative example of the limit area extension to a volume is shown in Figure 3.6. A 3D model with linear propagation profile is used as an initial model. The updates are made repeatedly around the equipotential curve $p_0(x) = p_M$, which describes a ring around the cone shaped model. The flattening effect observed in Figure 3.5 is again observed here, although in a 3D shape now. As with the 2D example, even with this deformation, the model does not cross a boundary defined by two surfaces. The space constrained between these two surfaces defines a limit volume. In the plot, the smaller cone limits the inner boundary, while the bigger cone, encapsulating all other surfaces, limits the outer boundary. The cuts in the plot, made only for visualisation purposes, reveal an identical shape to that in Figure 3.5.

**False initialisation** By relaxing the initialisation constraint, i.e., allowing an arbitrarily wrong initial model, it can be shown that with perfect localisation and sufficient iterations the model can be brought in finite time to the limit area.

An initial model can be defined, for example, as:

$$p_0(x) = B(x, p_{out} + \Delta p_{out}, \gamma + \Delta \gamma), \quad (3.30)$$

where $\Delta p_{out}$ and $\Delta \gamma$ are offsets of the output power and of the attenuation factor,
respectively. Considering $x_t$, there are again two measurement cases to be looked at:

1. As with perfect initialisation, if $x_M$ lies outside of the support of the weighting function $f_c$, no update is made.

2. Otherwise, the inequality $|p_{k+1}(x_t) - p_{M,k+1}| \leq |p_k(x_t) - p_{M,k+1}|$ still holds, but now $p_0(x_t)$ is arbitrarily wrong and a finite number of iterations is required in order to bring the feature map at $x_t$ into the limit area. In order to reach a feature map that is inside the limit area, all positions on the feature map must be sufficiently updated and therefore the measurements must be taken spaced at most by $\phi$, so that the updates cover the complete area.

Assuming that the initial model lies completely outside the limit area and that the considered fixed position $x_t$ is given by $x_M$, i.e., $x_t = x_M \forall k$, so that $p_M(x_t)$ is constant, then applying (3.30) on (3.10) at $x_t$ gives:

$$p_k(x_t) = B(x_t, p_{\text{out}} + \Delta p_{\text{out}}, \gamma + \Delta \gamma) \cdot (1 - F_k) + p_M(x_t) \cdot F_k$$

(3.31)

Due to the constant measurement positions $f_{c,k}(x_t) = \kappa \forall k$. Then, using the result of Theorem 3.2 and the exponential filter properties, demonstrated in Appendix B.2, the following relation holds:

$$F_k(x_t) = 1 - (1 - \kappa)^k$$

(3.32)

Hence, the influence of the initialisation will decrease according to $(1 - \kappa)^k$ and the measurement term will reach $p_M(x_t)$ according to $1 - (1 - \kappa)^k$. In finite time $p_k(x_t)$ will reach the bounded region defined by $p_{M+}$ and $p_{M-}$.

Assuming a linear propagation model, making $p_M(x_t) = p_{\text{out}} - \gamma \cdot x_t$, making $B(x_t, p_{\text{out}} + \Delta p_{\text{out}}, \gamma + \Delta \gamma) = p_{\text{out}} + \Delta p_{\text{out}} - (\gamma + \Delta \gamma) \cdot x_t$ and setting $p_k(x_t)$ at the bounded region as $p_{\text{out}} - \gamma x_t + \gamma \phi$, then substituting these terms in (3.31) the following inequality can be written:

$$(1 - \kappa)^k \leq \frac{\gamma \cdot \phi}{\Delta \gamma \cdot x_t - \Delta p_{\text{out}}},$$

(3.33)
which can be used to determine the required convergence time $k$ for this particular initialisation.

This analytical investigation shows that both SLL parameters should be small: $\phi$ should be small to enable a tight limit area, while $\kappa$ should be small to reduce noise effects. Otherwise, the larger the two parameters are, the faster (large $\kappa$) and wider (large $\phi$) the feature map learning is. A good trade-off between accuracy and speed could be achieved by starting with larger parameters and reducing them over time.

### 3.2.2.2 Statistical conditions

In this section the influence of real localisation on the performance of SLL is investigated. After defining real localisation, the statistical conditions for convergence towards the analytical bound are identified using simulations. The statistical conditions are confirmed by real world experiments.

**Real Localisation** When the localisation is not perfect, then the estimated position $x_c$, where the update function $f_c$ is centred, is not guaranteed anymore to be the measurement position $x_M$.

Now $x_{c,k+1}$ is estimated using the measurement $p_{M,k+1}$ and a NN search on $p_k(x)$, looking for the best $x = x_{c,k+1}$ for which $p_k(x_{c,k+1})$ is the closest match for $p_{M,k+1}$, as explained in chapter 2. For the simulations the NN algorithm is run on the quantised 1D space comprised in the interval $[0; L]$.

**Simulations** Some experiments were defined in order to reveal the statistical conditions that need to be satisfied to reliably obtain a good performance of SLL.

The experiments performed differ with respect to the distribution and the order of measurements. From the analytical considerations it is known that both SLL parameters, $\kappa$ and $\phi$, should be small at the end of the learning time to achieve a high accuracy and larger at the beginning to achieve a fast learning. To avoid a superposition of the statistical effects here investigated with effects possibly introduced by time-varying parameters, $\kappa$ and $\phi$ are set to fixed values and kept constant over time. Unless otherwise mentioned, the following characteristics are common to all experiments presented:
• The feature map is defined on the interval \([0; 15]\) given in metres. The distance \(\Delta x\) between consecutive positions on the discrete feature map is 0.1m.

• The shape of the weighting function \(f_c\) is defined as:

\[
    f_c(x) =\begin{cases} 
    \kappa \cdot \frac{\phi - |x - x_c|}{\phi}, & \text{if } |x - x_c| \leq \phi \\
    0, & \text{if } |x - x_c| > \phi
   \end{cases}
\]

(3.34)

• The learning parameter \(\kappa\) is set to 0.5 and \(\phi\) is set to 3m.

• The measurement positions \(x_M\) are uniformly distributed over space. The distance between measurement positions \(\Delta x_M\) as well as the sequence with which they are chosen are defined individually per experiment.

• The plots show the feature map at the initialisation (dashed lines) and at the end of simulation (bold lines), the measurements (thin line or circles) and the limit area (dash dotted lines), with power as a function of position.

• The plots also show the RMS feature map error \(e_{FM,k}\), defined as the RMS error between the true model defined by the measurements

\[
p_{true} = [p_{true,1}, \ldots, p_{true,q}]^T = [p_M(x_1), \ldots, p_M(x_q)]^T,
\]

and the actual feature map \(p_k(x)\) at a given instant \(k\):

\[
e_{FM,k} = \sqrt{\frac{1}{Q} \sum_{q=1}^{Q} (p_k(x_q) - p_{true}(x_q))^2},
\]

(3.36)

with \(Q\) as the number of reference positions on the radio map.

Other experiment characteristics are summarised in Table 3.1 at the end of this section. The main goal in presenting these simulation results is to show which statistically motivated constraints need to be fulfilled such that the determined analytical properties are achieved and a proper algorithm operation is obtained.

**Exp. 3.1.** Figure 3.7 depicts two complementary cases. The distance \(\Delta x_M\) between two succeeding measurement positions is 0.1m, which results in 150 measurements.
Figure 3.7: Exp. 3.1 - Feature map learning for sequentially ordered measurements. The distance between two succeeding measurement positions is smaller than ϕ.

ϕ is set to 1m, which is bigger than Δx_M. The initial radio map, as well as the measurements are given by the linear equation p_0(x) = p_{out} - γ · x, with p_{out} = -20 dBm, γ = 2 dBm/m (since the initial radio map and measurements are coincident, only the dashed line is plotted).

The measurements are sequentially ordered, taken either at increasing or decreasing coordinate values, i.e. from 0 to L or from L to 0. The feature map evolution in Figure 3.7a is given by the bold lines after the ordered sequence of measurements is cyclically used 70 times, which results in 10570 iterations.

The upper bold lines show the result when the order goes from 0 to L, while the lower bold lines show the result when the order goes from L to 0. A gray scale is used to differentiate the feature maps at distinct iteration steps k, together with an explicit labelling of the upper part, which applies to the lower part as well. The light gray lines (k = 140) show the feature map at the beginning of learning, and how they break through the boundary of the limit area, growing distant from the perfect initialisation at each completed cycle. Note that the slope of the final feature map depends directly on Δx_M.

The feature map error in Figure 3.7b are identical for both cases, since they are complementary experiments. e_{FM,k} departs from 0, since the initialisation is perfect and increases until a steady state is reached. The small oscillations visible in the radio
Figure 3.8: Exp. 3.2 - Feature map learning for sequentially ordered measurements. The distance between two successive measurement positions is equal to $\phi$

map error are caused by the periodicity of the measurement locations.

An important feature of SLL, called edge effect, is shown in this figure. Due to the sequence of narrowing close positions, almost the entire radio map lies outside the theoretical limit area. However, because of the limitation of the localisation space between 0 and $L$ together with the monotonicity of the propagation function, the strongest RSS will be learned at $x = 0$ and the weakest RSS will be learned at $x = L$, even for real localisation. If the localisation tries to locate a measurement outside the given interval, the estimated position is set to the corresponding edge.

Exp. 3.2. Figure 3.8 shows a slightly different setup as in Fig. 3.7, but again with complementary cases. Here, the distance between measurement positions $\Delta x_M$ has the same value as $\phi$, set to 3m, i.e., $x_M = \{0, 3, 6, 9, 12, 15\}$m, marked with circles on the plot. The initial feature map and the measurements follow the same linear propagation as in the last simulation, for which reason once again only the dashed line is shown. Each measurement is taken repeatedly at the same position 30 times before going to the next position, which gives 180 iterations.

The final feature map is given in Figure 3.8a by stair steps, with a spacing of $\phi$. The lower stair (bold black line) represents the sequence from 0 to $L$ and the upper stair (bold gray line) the sequence from $L$ to 0. The reason for the shape of the feature map is as follows. At the beginning the feature map is perfect and real localisation delivers the exact true position. After 30 updates at the same position $x_M$,
a step is formed, with wideness defined by $\phi$. However, even drifting away from the perfect initialisation, the feature map value at $x_M$ is still correct and thus consecutive measurements at this position are still correctly located. At the next measurement position the real localisation still delivers the exact match, since the feature map has not been changed at this position by the last update series. A new step is formed at this position, and one half of the previous step is entirely replaced by this new one while the other half remains.

The feature map error in Figure 3.8b for both cases are also identical, since both experiments are again complementary. $e_{FM,k}$ starts from 0 and rises, although the feature map for both cases remains inside the limit area this time. This flattening effect, which forms the stair steps, is a direct result of (3.14), with the replacement of the initial model by a term dependent on the measurements $p_M$. This experiment, although with real localisation, can also be used to understand how the limit area is determined in section 3.2.2.1.

Exp. 3.3. In Figure 3.9 the propagation profile is logarithmic, according to $p_M = p_{out} - \gamma \cdot 10 \log(\frac{\Delta x_M}{0.1})$, with $p_{out} = -20$dBm, and $\gamma = 2$dBm. For this experiment however, the uniform distribution is with respect to the measured power values ($\Delta p_M = 1.8$dBm) and not to the measurements positions, thus $\Delta x_M$ is not equally spaced. The initial feature map follows the measurement profile but with an offset $\Delta p_{out}$ of $-10$dBm.

The most important change with respect to the previous experiments is that the
Figure 3.10: Exp. 3.4 - Feature map learning for randomly ordered measurements. The initialisation shows an offset and the measurements are noisy.

25 measurement positions now are randomly selected during 1000 iterations. The final feature map in Figure 3.9a follows a linear interpolation between the strongest and the weakest RSS, as shown by the bold black line. This reflects the linearly spaced measurements in the power axis. In Figure 3.9b \( e_{FM,k} \) does not start from 0 anymore, since the offset \( \Delta p_{out} \) confers an initial error to the model. The final feature map diverges from the measurements, breaking through the limit area boundary.

**Exp. 3.4.** In Figure 3.10 \( \Delta x_M \) is 0.6m and the distribution of measurements is uniform with respect to the space and not anymore with respect to the power, as in Exp. 3.3. Figure 3.10a shows that measurements and the initial feature map follow the linear propagation as defined before, and the initialisation is done with an offset \( \Delta p_{out} = -10 \text{dBm} \), as in the previous experiment. A Gaussian noise with standard deviation of 7dBm was added to the measurements, which is shown by the error bars. This noise is larger than the inaccuracy caused by the limit area (defined by \( \phi = 3 \text{m} \)) which corresponds, in the considered linear model (\( \gamma = 2 \text{dBm/m} \)), to a maximum deviation of \( \gamma \cdot \phi = 6 \text{dBm} \). The 26 measurement positions are randomly selected during 1000 iterations.

Likewise in Exp. 3.3, \( e_{FM,k} \) does not start from 0, as Figure 3.10b shows. Also, it exponentially decreases as a result of the combination of two effects: the random order and the limit value, as in (3.14). Due to the averaging properties of SLL for \( \kappa < 1 \) the noise is significantly reduced and the final feature map, shown by the bold
black line, falls clearly inside the analytical boundary. It is important to note that a uniform distribution in space is truly a requirement. When the measurement positions are otherwise selected, the measurements themselves follow a distribution that does not correspond anymore to their real profile. This wrong distribution will feed the learn algorithm with false information that misleads the learning performance, as in Exp. 3.3. On the other hand, the form of the initialisation has no influence, as long as the initial feature map represents some physically possible propagation model, for example, maximum RSS value at the BS and decay of the RSS with growing distance.

Exp. 3.5. In the experiment shown in Figure 3.11 $\Delta x_M$ remains with 0.6m. The distribution of measurements is again uniform with respect to the space. Measurements follow a logarithmic propagation profile as defined before, but a wall with an attenuation factor of 10dBm is placed at 7.5m, which causes a discontinuity in the measurement profile, as displayed in Figure 3.11a. The initialisation of the feature map also follows the same logarithmic profile, but with an offset $\Delta p_{out}$ of $-20$dBm and ignoring the discontinuity. The final feature map lies again inside the limit area. The wall discontinuity is learned but also smoothed by the SLL.

Exp. 3.6. Figure 3.12 shows the last experiment in this subsection, performed with real data collected at a LOS scenario. This experiment is different from the previously described simulations because real world signal propagation is more complex and not necessarily linear but close to logarithmic as can be seen in Figure 3.12a. Therefore the
Figure 3.12: Exp. 3.6 - Feature map learning for real world data

radio map has been initialised according to a logarithmic propagation profile, but one which is still far from the true measurements such that there is improvement potential, as $e_{\text{FM},k}$ in Figure 3.12b shows.

Due to the findings in the previous experiments, SLL has been applied using a uniform distribution of measurement positions in combination with a random order. Measurements were taken with a spacing of $\Delta x_M = 0.6m$. Each of the 24 positions was measured 30 times, so that a representative collection of noisy measurements for each position was achieved, being then randomly selected during 1000 iterations.

The final feature map in Figure 3.12a (bold black line) shows that the real propagation profile, which is reasonably approximated to a logarithmic shape, is learned. The noise is significantly reduced by SLL, as the error bars show, such that the final feature map is within the limit area. The limit area boundaries follow the real measurement profile, still displaced by $-\phi$ and $+\phi$.

Hence, the identified statistical conditions, as the uniform distribution in space and the random ordering of measurements, were verified by this real world experiment. Noteworthy is that the SLL learns the (averaged) measurements and maps them on the localisation area between 0 and $L$. It is important that all the measurements are seen, and that they are equally probable with respect to the space domain. Table 3.1 summarises the main differences between the experimental setups.

50
3.3 Comparison between SLL and SOM

There is a striking resemblance between (3.2) and (3.5). In fact, the update law for SLL is the same as for SOM. However, the SOMs exist over a discrete lattice with a finite number of neurons, whereas the SLL can work in a continuous space (in which case the concept for a lattice vanishes, since there can be infinite neurons over a closed interval). However, a space discretisation for the SLL is the usual choice for implementation, defining the positions in the feature map where the continuous map is sampled. The neighbourhood function $h_{ci}$ becomes the weighting function $f_{c,k}$, the model $m_i$ becomes the radio map $p_k(x)$, and the input vector $\xi$ becomes the measurement vector $p_M$.

In the following, each of the learning steps of the SOM are related to their equivalent at the SLL as well as the two phases of the adaptive process.

3.3.1 Initialisation

The weights $m_i$ in the SOM neurons are also referred to as models. They can be randomly initialised or given some value if any a priori information about the signal distribution is known. If the SOM already starts in a somewhat organised state, then the learning and subsequent convergence can be achieved much faster [59,60].

The initialisation is critical for the SLL to work, since it dictates the initial accuracy of the localisation system. The SLL must start with a coarse initial model, which at least presents some plausible physical property (for example, a feature map where the maximum RSS is placed at the BS position and decays with increasing distance). This ensures that the initial model can be used for localisation queries, even if it is not very

<table>
<thead>
<tr>
<th>Exp.</th>
<th>$\Delta x_M$</th>
<th>$\phi$</th>
<th>Sequence</th>
<th>Source</th>
<th>Initialisation</th>
<th>Investigated aspect</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>0.1</td>
<td>1</td>
<td>${0 \rightarrow L \ 0 \leftarrow L}$</td>
<td>Simulation</td>
<td>Perfect</td>
<td>Edge effect</td>
</tr>
<tr>
<td>3.2</td>
<td>3</td>
<td>3</td>
<td>${0 \rightarrow L \ 0 \leftarrow L}$</td>
<td>Simulation</td>
<td>Perfect</td>
<td>Limit Value, Limit Area</td>
</tr>
<tr>
<td>3.3</td>
<td></td>
<td>3</td>
<td>Random</td>
<td>Simulation</td>
<td>$\Delta p_{out}$</td>
<td>Uniform power distribution, $\Delta p = 1.8$dBm</td>
</tr>
<tr>
<td>3.4</td>
<td>0.6</td>
<td>3</td>
<td>Random</td>
<td>Simulation</td>
<td>$\Delta p_{out}$</td>
<td>Uniform position distribution, Noise reduction</td>
</tr>
<tr>
<td>3.5</td>
<td>0.6</td>
<td>3</td>
<td>Random</td>
<td>Simulation</td>
<td>$\Delta p_{out}$</td>
<td>Discontinuity</td>
</tr>
<tr>
<td>3.6</td>
<td>0.6</td>
<td>3</td>
<td>Random</td>
<td>Real</td>
<td>$\Delta p_{out} + \Delta \gamma$</td>
<td>Real world performance</td>
</tr>
</tbody>
</table>

Table 3.1: SLL Simulation Characteristics
accurate. This initial model represents an already ordered state, as the feature map is never initialised with random values.

3.3.2 Competition

The competition in SOM aims to identify a winning neuron $c$ for a given input signal $\xi$. With SLL the competition step occurs exactly with the localisation query, where a position in the feature map must be retrieved as the best match for a given measurement $p_M$ and the actual radio map $p_k(x)$. This search with both SOMs and SLL is normally accomplished using a NN search with some distance measure in the input space as cost function. If the SLL map is defined continuously then the position estimate is done with a minimum search with gradients. The winning neuron at SLL has a fixed position in a reference frame and this coordinate is returned as the answer to the location query. Nevertheless, the position estimate is not constrained to the sampling positions, i.e., the actual neurons coordinates in the reference frame: for example, if the location estimate is retrieved with kNN.

The localisation has also an important role for the SLL. Since it determines the location where the update will be made, if the localisation delivers a $x_c$ very far from the (usually) unknown measurement position, the SLL will fail to improve the initial model.

3.3.3 Cooperation

The cooperation defines a neighbourhood area in the latent space where the update is made. The size of this area and how the update is weighted is defined by $h_{c,i}$ for SOMs and by $f_{c,k}$ for SLL. The subscript $c$ indicates that these functions are centred at the neuron/position delivered by the competition/localisation step.

The SLL defines the amplitude and wideness of the update area with the control variables $\kappa$ and $\phi$, respectively. This finds its equivalent with SOM as (3.1) shows with $\alpha(k)$ and $\sigma(k)$. However, for SOMs the neighbouring neurons selected for cooperating with the winning neuron can be chosen explicitly, as a fixed number of neighbours in the lattice (for example, two neighbours for 1D lattices, and four or even eight for 2D squared lattices).
For both SOMs and SLL the update amplitude is normalised, that is, it is always smaller than one. In this way, the model update as defined in (3.2) and (3.5) will be never greater than the difference between the weights at the selected neuron/position and the new input pattern.

The shape of $h_{ci}$ is usually free, but for most proofs in 1D (see [11, 29, 30]) a constant update is chosen in a given step $k$, and that for at most two neighbours. For SLL, the requirement that $f_{c,k}$ has its maximum at $x_c$ means that the localisation step is trusted and that the update should reflect that, favouring the position $x_c$ with the bigger update in the direction of the measurement and penalising the neighbours with smaller updates. The wideness of $f_{c,k}$ also reflects the trust given to the localisation step. Therefore, a wider area chosen at the beginning means there is little trust in the localisation. With enough iterations the model is improved and therefore it can be trusted more, which allows the update area to be narrower. The limit case, where the area is so narrow that just the centre position $x_c$ is updated, corresponds to the 0-neighbour configuration of SOM, which is a type of vector quantisation (VQ) [30] and is of no interest for the SLL.

### 3.3.4 Adaptation

The adaptation occurs according to (3.2) and (3.5), inside the update area defined at the previous step. The size of the adapted neighbourhood and the magnitude of this adaptation is defined by $h_{ci}$ and $f_{c,k}$ as explained for the cooperation phase.

For the SLL the adaptation aims to bring the feature map in the direction of the measurements, even if the localisation is not perfect. In section 3.2.2.2 it has been shown that under certain conditions the adaptation has a flattening effect on the model, which indicates the need for statistical variation of the input patterns. They must be measured with a uniform distribution of positions in the area where the map must be learned.

### 3.3.5 Self Organisation

During the learning, the self organising phase is the most critical for the SOMs, especially if the model has been randomly initialised. The definition of self organisation is
somewhat subjective, and mathematically it has only been defined for 1D SOMs, allowing for mathematical proofs about self organisation for 1D SOMs to be found [20,33–35]. While the definition of a self organising state for 1D is quite trivial (an ascending or descending monotonic ordering of scalar values), the same cannot be said about a definition for 2D maps or higher dimensions. Most of the simple illustrative examples with 2D maps found in the literature show a square network and the input space is also 2D (in a way that the weights represent the coordinates of the neurons) [51,60]. Such examples intuitively show that the self organisation can be visualised as the “spreading” of a “towel”, that begins with disordered “folds” due to the random initialisation and goes to a “spread” and “open” square shape after the self organisation. What is usually not mentioned in these examples is that this intuitive organised state is not unique. In fact, for this configuration there are 8 possible states that result in this “spread” towel (due to rotation and permutation of neuron weights in the latent space coordinates).

This phase is of little interest for the SLL, since it already starts in an organised state. The initialisation of the radio map, though coarsely made, defines one of the possible organised states (in the sense of SOMs). However not all of the possible organised states are of interest to the SLL. In fact, the real propagation profile belongs to an unique organised state, which may mismatch the state defined by the initial radio map. The transition from one state to another is covered by the SLL algorithm.

Figure 3.13 shows illustrative examples of these 8 possible states. In Figure 3.13(a) one possible SLL scenario is constructed: a BS, placed at the origin, spreads radio signals over a rectangular grid with 9 nodes. Figures 3.13(b) to 3.13(i) show all 8 possible self organising states according to SOM, with the filled circle representing the BS position in the lattice. Nevertheless, for SLL the states in Figs. 3.13(b) and (g) are admissible, matching the position of the BS.

### 3.3.6 Convergence

The convergence phase takes place once a self organising state is found, making a fine tuning of the neuron weights in order to best model the input patterns. During the convergence, the neighbourhood area as well as the update amplitude are made smaller.
The mathematical definition of convergence in 1D SOMs is not as uniform as the definition of self organisation, but many authors have successfully proven almost sure convergence, that is, with probability 1, considering continuous time and taking the solutions of a system of ODEs [11, 20, 36, 90]. A basic convergence definition says that once a self organised state is achieved, the system never leaves this state, i.e., an absorbing state.

Once again, all the proofs remain in the 1D domain. A naive approach is to make \( h_{ci} = 0 \), so that the system “freezes” since from (3.2), \( m_i(k + 1) = m_i(k), \forall k \). For higher dimensions, even with simpler assumptions, the proofs point to instabilities, and some even suggest the correspondence to an irreducible Markov chain, i.e., the probability of going from one state to any other is always positive. Due to that, in a practical scenario with several BSs, the SLL presents no absorbing state.

Such a property is very dangerous for SLL, for it means that even starting in a matching state (and the initial model can assure that), there is a chance that the system is pushed to another state, which poorly describes the propagation, therefore destroying the feature map. In fact, Figures 3.7 and 3.8 in section 3.2.2.2 show particular sequences of input patterns that can damage the feature map instead of improving it.
The solution to this problem relies on the use of fixed points, as suggested for the original SOM [60]. The fixed points work as supervised iterations while the unsupervised learning takes place. Inputs patterns that are previously known to belong to a specific cluster can be used to force the right neuron to match it, such that some updates are made in the right neighbourhood.

In SLL the fixed points are the labelled samples that are normally used for calibration of localisation systems. There is no need to take too many fixed points as this would contradict the main advantage of SLL, which is the relinquishment of calibration data. The mutual measurements of BSs are a reasonable and efficient option to provide such fixed points, since their locations are known and there is no need for a technician to take measurements by hand. Returning to the analogy of a “towel”, the fixed points behave like “nails” clamping the “towel” to a fixed frame.

3.4 SLL: Real World Experiments

Some of the proofs so far have only been shown for one-dimensional examples for the same reason as the corresponding proofs for SOM: they are unfeasible for more dimensions, although there have always been practical examples in 2 or 3 dimensions that do converge. The SLL was tested in real world 2D scenarios with successful results and this validates the algorithm beyond a theoretical set of 1D feasible proofs.

The SLL is independent of the radio technology used, and to show this, examples using DECT and WLAN networks are presented. The NN is applied to localisation, as described in Sect. 2.3.1, and the SLL is compared between two different initial models, using the same validation data.

The localisation error $e_{pos,k}$ is calculated as:

$$e_{pos,k} = \frac{1}{Q} \sum_{q=1}^{Q} \sqrt{(x_{k,q} - x_q)(x_{k,q} - x_q)^T},$$

(3.37)

with $x_q$ as the known validation position and $x_{k,q}$ as the estimated position using the radio map at time $k$, and the validation measurement $p_{\text{validation},q}$.

A RMS feature map error is also evaluated, in the same way as in (3.36), but now
with \( N \) BSs:
\[
\epsilon_{FM,k} = \sqrt{\frac{1}{Q \cdot N} \sum_{q=1}^{Q} \sum_{n=1}^{N} (p_{true,q,n} - p_{k,q,n})^2}, \tag{3.38}
\]
with \( p_{k,q,n} \) as the radio map evaluated at time \( k \), position \( x_q \) and BS \( n \).

The true model \( p_{true,q,n} \) at position \( x_q \) for BS \( n \) is defined as:
\[
p_{true,q} = \begin{bmatrix}
p_{true,q,1} \\
\vdots \\
p_{true,q,N}
\end{bmatrix} = \begin{bmatrix}
\text{mean}\{p_{training,q,1}\} \\
\vdots \\
\text{mean}\{p_{training,q,N}\}
\end{bmatrix}, \tag{3.39}
\]
with \( \text{mean}\{p_{training,q,n}\} \) as the mean value over the whole set of training data at the position \( x_q \) for BS \( n \).

The weighting function \( f_{c,k} \) used for the 2 dimensional SLL has the form:
\[
f_{c,k}(x) = \begin{cases} 
\kappa \cdot \left(1 - \frac{d_{c,k}(x)}{\phi}\right), & \text{if } d_{c,k}(x) \leq \phi \\
0, & \text{if } d_{c,k}(x) > \phi 
\end{cases} \tag{3.40}
\]
with \( d_{c,k}(x) = \sqrt{(x - x_{c,k})^T(x - x_{c,k})} \), whose shape corresponds to Fig. 3.3b.

**Exp. 3.7** (Office Environment - DECT). The first test environment is an office environment with 9 DECT BSs installed (marked as squares), as depicted in Fig. 3.14. There are 223 training positions (crosses) which are also used as validation points. The validation measurement set is constant in time, with only one measurement for each position. Instead, a particular position may have more measurements in the training set, which are randomly selected. To avoid superposition of different effects, the control parameters \( \kappa \) and \( \phi \) were made constant during the simulation.

Considering the DECT office environment, the 223 training positions were randomly selected during 5000 iterations, and a corresponding measurement was also randomly selected and used as unlabelled input data for the SLL. The initial models considered were the RM and the DPM, as described in Sect. 3.2.1. The control parameters were set to \( \kappa = 0.1 \) and \( \phi = 10 \text{m} \).

Figure 3.15a shows the evolution of SLL for 5000 iterations, displaying \( \epsilon_{pos,k} \), and in Fig. 3.15b \( \epsilon_{FM,k} \). From these plots it can be stated that the linear model is simpler than
Figure 3.14: Office environment with DECT BSs

The DPM as the initial error is bigger at first. In fact, the DPM is more complex than the RM, since environment layout information is built into the model. Nevertheless, the SLL steps improve both models, as can be seen with the fall of both errors. The two plots also indicate the direct relationship between \( e_{\text{pos},k} \) and \( e_{\text{FM},k} \).

A relative position density measure is defined as:

\[
\rho_{\text{pos}} = \frac{\sqrt{A}}{\sqrt{Q - 1}},
\]

(3.41)

where \( A \) is the feature map covered area and \( Q \) the number of feature map positions. In this way, a squared area with 1m side and 4 points, one at each vertex, has density 1m, meaning that the positions have a spacing of 1m from one another.

In Table 3.2 the main parameters for this experiment setup are summarised (facility area, number of positions “Nr Pos”, position density \( \rho_{\text{pos}} \), \( \kappa \) and \( \phi \), as well as initial and final value of \( e_{\text{pos},k} \), and \( e_{\text{FM},k} \), and the percentage improvement achieved after 5000 iterations.

**Exp. 3.8 (Office Environment - WLAN).** The second test environment is also an office environment with 14 WLAN BSs installed (squares), as seen in Fig. 3.16. There are 114 training positions (crosses) used also as validation points. For this office environment, the same procedure as for the DECT setup was used: 114 training positions were randomly selected and their measurements, also randomly selected, were used as
unlabelled input data during 5000 iterations. The initial models were again the RM and the DPM. The control parameters were set to $\kappa = 0.1$ and $\phi = 5$ m.

Figures 3.17a and 3.17b show the results of SLL for this setup. Here, the initial error difference between the linear model and the DPM is even bigger. This can be explained by the fact that the walls of this office are covered with metallic foils that highly attenuate the signals. Accordingly, the information about the walls results in a very good initial model, leading to low localisation error prior to the start of learning. Further learning by the SLL brings improvement, as can be seen with the fall of both errors.
Figure 3.17: WLAN office environment (a) localisation error and (b) feature map error

The results for the WLAN setup are also presented at Table 3.2, together with the results for the DECT setup.

<table>
<thead>
<tr>
<th>Exp.</th>
<th>3.7 (DECT)</th>
<th>3.8 (WLAN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area (m²)</td>
<td>4517.75</td>
<td>1241.46</td>
</tr>
<tr>
<td>FM Positions</td>
<td>223</td>
<td>114</td>
</tr>
<tr>
<td>( \rho_{\text{pos}} ) (m)</td>
<td>4.83</td>
<td>3.64</td>
</tr>
<tr>
<td>( \kappa )</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>( \phi ) (m)</td>
<td>10</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>error</th>
<th>( \varepsilon_{\text{pos}} ) (m)</th>
<th>( \varepsilon_{\text{FM}} ) (dBm)</th>
<th>( \varepsilon_{\text{pos}} ) (m)</th>
<th>( \varepsilon_{\text{FM}} ) (dBm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>RM</td>
<td>7.06</td>
<td>6.21</td>
<td>7.51</td>
</tr>
<tr>
<td></td>
<td>DPM</td>
<td>7.51</td>
<td>5.99</td>
<td>10.70</td>
</tr>
<tr>
<td>5000</td>
<td>RM</td>
<td>4.84</td>
<td>5.06</td>
<td>4.61</td>
</tr>
<tr>
<td></td>
<td>DPM</td>
<td>4.71</td>
<td>4.68</td>
<td>6.97</td>
</tr>
</tbody>
</table>

| improvement (%) | 31.42 | 18.41 | 38.64 | 21.30 |
| | 34.90 | 16.61 | 39.66 | 27.94 |

Table 3.2: SLL Real World Experiment Results
Chapter 4

SPLL: Simultaneous Probabilistic Localisation and Learning

4.1 SPLL: Main Algorithm

The Simultaneous Probabilistic Localisation and Learning (SPLL) is an iterative algorithm based on the improvement of a feature map using unlabelled samples as its predecessor, the SLL. The feature recorded in the SPLL feature map is a probability density function (pdf), in contrast with only the mean RSS value in the SLL feature map. A new measurement is used to locate a user with probabilistic localisation, and at the estimated location an update is made on the feature map, as well as at neighbouring positions. The probabilistic localisation algorithm used is the MMSE due to its robustness as explained in section 2.3.4. The pdf is a natural choice for a description of a continuous random variable. However, as a discretisation step is invariably introduced when implementing the algorithm, the formally correct description should be of a probability mass function (pmf) for discrete random variables. Throughout this thesis, the description of the algorithm will use pdfs as a base, and pmfs for the implementation.

The following sections introduce the main aspects that together form the SPLL, concluding with the algorithm itself.
4.1.1 Non Parametric Density Estimation

In order to estimate a position with MMSE, \( \Pr(p_M|x) \) must be evaluated in some way so that (2.6) can be calculated. If \( g(p,x) \) is a pdf, then \( \Pr(p_M|x) \) is the integral over the volume of \( g(p,x) \), with integrand limits defined by \( p_M \) and some small tolerance. If \( g(p,x) \) is a pmf, then \( \Pr(p_M|x) \) is evaluated directly as \( g(p_M,x) \).

To build \( g(p,x) \) from the measurements there are many possibilities to consider: parametric approaches are often found in the literature under the assumption that the real distribution can be approximated by some known model, usually describing the pdfs in terms of Gaussian distributions or even Gaussian mixtures. Non-parametric approaches, such as histogram and kernel based methods, are then used when the pdf cannot be described in terms of a set of parameters.

While the histogram based estimation always delivers a discontinuous solution, the kernel based, known also as Parzen method \([82, 88]\), can provide a smooth and continuous solution with an appropriate kernel choice.

Considering only one BS and a particular position \( x_q \) in the feature map, for a given set of observations \( \{p_{M,1}, \cdots, p_{M,W}\} \), an estimate of a density function in one dimension with the kernel based method, described thoroughly in \([100]\), is:

\[
g_q(p) = \frac{1}{W \cdot h} \sum_{i=1}^{W} K\left(\frac{p - p_{M,i}}{h}\right),
\]

where \( K(z) \) is the kernel function, and \( h \) a smoothing parameter. \( K(z) \) is often regarded as a pdf, when \( K(z) \geq 0 \) and \( \int K(z) \, dz = 1 \). In \([82, 100]\) a list with common kernel functions for univariate data is available. \( K(z) \) may have almost any shape, but the choice of the kernel directly influences the estimated pdf. For example, a rectangular kernel, with discontinuity at its borders, will also generate a discontinuous pdf.

The expression in (4.1) works as a batch. That is, all observations must be available in order to calculate the sum. Since the learning is described as an iterative process, an iterative formulation for (4.1) would be more useful.

Let \( h = 1 \), and let the original notation of the kernel function be changed from \( K(p,p_{M,i}) \) to \( f(p,p_{M,i}) = f_i \). Then (4.1) is rewritten as:
The next theorem can be stated.

**Theorem 4.1.** Consider the closed form of the kernel based density estimation in (4.2) and let the constraint that \( \int K(z) \, dz = 1 \) be relaxed, such that \( \int K(z) \, dz = \int f_i \, dp = S_i \).

Then a recursive formulation for (4.2) is:

\[
g_{k+1,q}(p) = g_{k,q}(p) \cdot \left( \sum_{i=0}^{k} S_i \right) + f_{k+1}(p) \sum_{i=0}^{k+1} S_i
\]

(4.3)

**Proof.** For the sake of readability, the dependencies of \( g \) and \( f \) on \( p \) and \( x_q \) are not explicitly stated. Let \( g_0 \) be the initial pdf such that \( \int g_0 \, dp = 1 \), then the first iteration is given by:

\[
g_1 = \frac{g_0 + f_1}{1 + S_1}
\]

and the second by:

\[
g_2 = \frac{g_1 \cdot (1 + S_1) + f_2}{1 + S_1 + S_2} = \frac{g_0 + f_1 + f_2}{1 + S_1 + S_2},
\]

where it can be verified that at this step the normalisation factor of \( g_1 \) is cancelled by the multiplication of \( (1 + S_1) \) and replaced by a new factor where the actual \( S_k = S_2 \) is taken into account.

Consequently, the term at \( k + 1 \) is:

\[
g_{k+1} = \frac{g_k \cdot \left( 1 + \sum_{i=1}^{k} S_i \right) + f_{k+1}}{1 + \sum_{i=1}^{k+1} S_i} = \frac{g_0 + \sum_{i=1}^{k+1} f_i}{1 + \sum_{i=1}^{k+1} S_i}
\]

(4.4)

If \( g_0 = 0 \) then (4.4) reduces to:

\[
g_k = \frac{\sum_{i=1}^{k} f_i}{\sum_{i=1}^{k} S_i}
\]

63
And if \( S_i = 1 \forall i \) then,

\[
g_k = \frac{1}{k} \sum_{i=1}^{k} f_i, \tag{4.5}
\]

which is the initial kernel method as in (4.2).

Considering the index \( k \) as a time measure, the kernel based estimation takes all contributions from 1 to \( k \) equally. Permutations on the sequence of observations has no effect at all at the estimated density since the sum is commutative.

Therefore, even with an iterative description, the kernel method is not adequate for learning yet. With SLL the newer observations (measurements) have a stronger effect than older ones, as section 3.2.2.1 shows. This “forgetting” effect is an important feature for the calibration algorithms here developed, since the initial model is not accurate and should be “forgotten” when a better model is learnt. Thus, a small modification in the recursive form of (4.4) is proposed in order to achieve this effect with the kernel method:

\[
g_{k+1,q}(p) = \frac{g_{k,q}(p) + f_{k+1}(p)}{\int \left( g_{k,q}(p) + f_{k+1}(p) \right) dp}, \tag{4.6}
\]

where the integral at the denominator normalises the term at \( k + 1 \) so that the area is equal to 1, which, combined with the fact that \( g_{k,q}(p) \geq 0 \), assures that (4.6) describes in fact a pdf.

The kernel at time \( k \) is centred at \( p_{M,k} \) and for the SPLL a natural shape choice was the Gaussian function:

\[
f_k = \kappa \cdot \exp \left( - \left( \frac{p - p_{M,k}}{\psi} \right)^2 \right), \tag{4.7}
\]

where \( \kappa \) controls the amplitude and \( \psi \) the width of the kernel. As defined for the SLL, \( \kappa \) is the learn rate and \( \psi \) corresponds to the smoothing parameter \( h \) from (4.1). In fact, \( \psi \) controls how smooth the learned pdf will be: if \( \psi \) is too wide, then the pdf will be very smooth and fine details such as peaks in the pdf will never be learned. On the other hand, if \( \psi \) is too narrow the pdf will be very jagged and rough. The condition imposed for the kernel in [100], forcing it to have unity area, is again relaxed since
(4.6) has already a normalising factor. Note that $f_k$ at SPLL is a function of power $p$ and not anymore of position $x_q$ as $f_{c,k}$ with SLL.

The next theorem shows the effect of the modification of the kernel method introduced in (4.6):

**Theorem 4.2.** Replacing the denominator of (4.6) with $A_{k+1}$ for the sake of readability, it is possible to write the recursive formula from (4.6) in closed form:

$$g_{k,q}(p) = \frac{g_{0,q}(p)}{A_{k+1}} + \sum_{i=1}^{k} \frac{f_i(p)}{A_{k+1}}\prod_{j=1}^{i-1} A_j \prod_{j=i}^{k} A_j$$

(4.8)

where $g_{0,q}(p)$ is the pdf chosen for the system initialisation at time $k = 0$ and position $x_q$.

**Proof.** For the sake of readability the dependencies of $g$ and $f$ on $p$ and $x_q$ are not explicitly stated. The first and second iterations of SPLL can be directly written as:

$$g_1 = \frac{g_0 + f_1}{A_1},$$
$$g_2 = \frac{g_0 + f_1 + f_2}{A_1 \cdot A_2},$$

which conform with (4.6).

Assuming that the closed form of the update law (4.8) at step $k$ is true, an inductive proof is constructed applying (4.8) on the recursive form (4.6) at step $k + 1$:

$$g_{k+1} = \frac{g_k + f_{k+1}}{A_{k+1}} = \frac{g_k}{A_{k+1}} + \sum_{i=1}^{k} \frac{f_i}{A_{k+1}}\prod_{j=1}^{i-1} A_j \prod_{j=i}^{k} A_j = \frac{g_0}{A_{k+1}\prod_{j=1}^{k} A_j} + \frac{1}{A_{k+1}} \sum_{i=1}^{k} \frac{f_i}{A_{k+1}}\prod_{j=1}^{i-1} A_j \prod_{j=i}^{k} A_j + \frac{f_{k+1}}{A_{k+1}}$$

65
Moving $A_{k+1}$ inside the product of the first and second fractions gives:

$$g_{k+1} = \frac{g_0}{\prod_{j=1}^{k+1} A_j} + \sum_{i=1}^{k} \frac{f_i}{\prod_{j=i}^{k+1} A_j} + \frac{f_{k+1}}{A_{k+1}}.$$  

and finally the last fraction can be moved into the sum so that the expression for step $k + 1$ is obtained in the same form as the expression for step $k$, which concludes the induction.

$$g_{k+1} = \frac{g_0}{\prod_{j=1}^{k+1} A_j} + \sum_{i=1}^{k+1} \frac{f_i}{\prod_{j=i}^{k+1} A_j}$$

The area defined by $A_{k+1}$ is composed of two terms, as seen in (4.6): the area under the pdf $g_{k,q}(p)$, which was normalised in the previous step $k - 1$, and consequently has a value of 1, and the area of the actual $f_{k+1}(p)$. Hence, $A_k > 1 \forall k$.

**Limit Value** Making $k \rightarrow \infty$ it is possible to observe what happens with (4.8) when the number of iterations grows. As $A_k > 1$ it follows that $\lim_{k \rightarrow \infty} \prod_{j=1}^{k} A_j = \infty$, which cancels the first term of (4.8). As a consequence, $g_k(p, x)$, as given in (4.8), tends to:

$$\lim_{k \rightarrow \infty} g_{k,q}(p) = \lim_{k \rightarrow \infty} \frac{g_0}{\prod_{j=1}^{k} A_j} \prod_{i=1}^{k} \frac{f_i(p)}{\prod_{j=i}^{k} A_j} = \lim_{k \rightarrow \infty} \frac{f_k(p)}{\prod_{j=1}^{k} A_j}$$

showing that the initial model $g_{0,q}(p)$ is entirely replaced by the second term in (4.8), which depends only on the measurements. The “forgetting” property is therefore verified since older terms of $f_i$ have less weight on the sum than the newer ones. This capability to “forget” old values while learning new ones is a necessary condition for the SPLL to work and is the major difference between the non-parametric density estimation developed here and the kernel methods found in the literature, where all measurements are equally weighted to compose the pdf.
4.1.2 Initialisation

As with the SLL initialisation in section 3.2.1, the SPLL initialisation must also comply with some physical plausibility, so that location queries can be performed right after system start.

Since the feature for the SPLL is a whole pdf on the RSS, counterpoising the mean RSS value used for the SLL, the complexity of the initial model $g_{0,q}(p)$ is inevitably greater than that of $p_{0}(x)$.

Still considering the initial model as a radial function around each BS, and the RSS mean value as a function with negative gradient for growing distance, the pdfs are chosen to be unimodal symmetric distributions using the distance dependent RSS mean values as the peak for those distributions.

Even constraining the decay shape of the RSS mean value, there are still infinite possible unimodal distributions with symmetry around the mean, and each of them with one or more parameters to control their individual shapes. As the initial model should be the simplest possible while still preserving plausible characteristics from physics, a natural choice for a distribution is the Gaussian distribution:

$$g_{0,q,n}(p) = \frac{1}{\sigma \sqrt{2\pi}} \cdot \exp \left( -\frac{(p - \mu(d_{q,n}))}{2\sigma^2} \right) \quad (4.10)$$

where $\mu(d_{q,n})$ can be obtained using (3.3) for example. $d_{q,n}$ is the relative distance from a position $x_q$ on the feature map to the BS$_n$.

However, a poor parameter choice for the Gaussian distribution can make the algorithm fail to learn since it operates blindly over the actual feature map. For example, if $\sigma$ is too big, then the Gaussian bell curves will be too wide, and in the limit the distribution approximates a uniform distribution over the interval where the feature map is defined, therefore making every position equiprobable. If $\sigma$ is too small, then the bell curves tend to Dirac impulses and only perfect measurement matches will return non null results. A reasonable dimensioning of $\sigma$ gives values comparable to the expected measurement noise.
4.1.3 The Learning Phase

Once the new density estimation is obtained, based on the kernel method but with the modification introduced at (4.6), and with the initial model \( g_{0,q,n}(p) \), the learning phase can begin. The SPLL works in the same way as the SLL. That is, a location query introduces a new measurement, and the delivered location estimate is used as the centre for an update in the feature map. The next subsections explain in detail the learning steps.

4.1.3.1 Localisation

The SPLL uses MMSE for localisation, as explained in section 2.3.4. With a new measurement \( p_M \), an estimate for location \( x \) is retrieved using (2.11). \( \Pr(x_q|p_M) \) is retrieved by a look up on the feature map. At each position \( x_q \) the actual measurement \( p_M \) is applied on \( g_q(p) \) so that \( \Pr(p_M|x_q) \) is retrieved as \( g_q(p_M) \), if \( g \) describes a pmf.

If there is no a priori information about where is the actual user location, then \( \Pr(x_q) = \frac{1}{Q} \), meaning that all positions on the feature map are equiprobable.

Then (2.6) can be written as

\[
\Pr(x_q|p_M) = \frac{\Pr(p_M|x_q) \cdot \Pr(x)}{\sum_{i=1}^{Q} \Pr(p_M|x_i) \cdot \Pr(x_i)} = \frac{g_q(p_M) \cdot \frac{1}{Q}}{\sum_{i=1}^{Q} g_i(p_M) \cdot \frac{1}{Q}} = \frac{g_q(p_M)}{\sum_{i=1}^{Q} g_i(p_M)},
\]

and directly applied to (2.11) to retrieve \( x \).

Noteworthy is that due to MMSE the location estimates always lie inside the geometrical boundaries defined by the feature map positions \( x_q \)s.

4.1.3.2 Neighbour Selection

Once a location estimate \( x \) is retrieved, then it must be selected which positions in the feature map will be effectively updated.

The position \( x \) is not necessarily coincident with some \( x_q \) due to the implicit averaging of several \( x_q \)s by the MMSE. However, even though \( x \) may not belong to the set of discretisation points that define the feature map, \( x \) is used as the centre of the
update area, defined as a circle with radius \( \phi \). \( \phi \) in SPLL has the same meaning as \( \phi \) defined for the weighting function \( f_c \) in SLL and depicted in Fig. 3.3.

Defining the Euclidean distance between the location estimate \( \mathbf{x} \) and the \( q^{th} \) feature map position \( \mathbf{x}_q \) as:

\[
d_q = \sqrt{(\mathbf{x} - \mathbf{x}_q)^T(\mathbf{x} - \mathbf{x}_q)},
\]

then every position \( \mathbf{x}_q \) for which \( d_q < \phi \) will be updated.

### 4.1.3.3 Feature Map Update

The feature map is updated at the \( \mathbf{x}_q \) positions inside the \( \phi \) radius. The update uses the non-parametric density estimation as explained in subsection 4.1.1. Considering the discrete time as \( k + 1 \), then the update law follows (4.6) applied at \( \mathbf{x}_q \).

Since the update is made at several positions using the same measurement \( \mathbf{p}_M \), a small modification is made at \( f \) in (4.7) so that the positions \( \mathbf{x}_q \) closer to the centre \( \mathbf{x} \) are favoured, with a higher update amplitude. In SLL, \( \kappa \) and \( \phi \) were decoupled variables that controlled the amplitude and geometric wideness of \( f_c \). Now from (4.7) the SPLL Kernel function \( f \) has its wideness in power scala controlled by \( \psi \), while \( \kappa \) still controls the amplitude. In order to impose a variable update dependent on the spatial geometry, \( \kappa \) is made dependent on \( d_q \) and \( \phi \), so that:

\[
\kappa = \kappa(d_q, \phi) = \begin{cases} 
\frac{\phi - d_q}{\phi} \cdot \kappa_{\text{max}}, & \text{if } d_q \leq \phi \\
0, & \text{if } d_q > \phi
\end{cases},
\]

where \( \kappa_{\text{max}} \) is the maximum value for \( \kappa \).

The control parameters of SPLL \( \kappa_{\text{max}}, \phi, \) and \( \psi \) can and should be made variable with time. Right after initialisation, they should start with large magnitude such that the model is brought from its false initialisation condition to the vicinity of the true model, and then the parameters should be made smaller such that the fine details of the model can also be learned.
4.1.4 Simulated Examples

In order to show the learning capabilities of the SPLL, some test scenarios are presented in 1D and 2D examples.

4.1.4.1 1D Test Bed

\( x_q = q - 1 \) with \( q = \{1, \cdots, 21\} \) in metres. Only one BS is considered and placed at \( x_1 = 0 \text{m} \). The propagation profile is Gaussian with standard deviation \( \sigma = 5 \text{dBm} \) and the mean \( \mu \) decays linearly with distance, such that \( \mu_q = P_{\text{out}} - \gamma \cdot x_q \), where the output power \( P_{\text{out}} = -20 \text{dBm} \) and the attenuation factor \( \gamma = 2 \text{dBm/m} \). Additionally, at \( x_{11} = 10 \text{m} \) a discontinuity of \(-20 \text{dBm}\) is placed, which models the effect of a thick wall or door. This discontinuity will be explored in different ways, defined individually for each shown example. The pdfs are recorded as pmfs in a discrete space with the RSS range from 0 to \(-100 \text{dBm}\) in steps of 1dBm. In this way, the numeric approximation of the pdf by rectangles with height \( g(p) \) and width 1dBm returns the same value as the pmf evaluated at \( p \). For each position, 1000 samples were generated from the correspondent distributions and used as measurements.

The initial model for the feature map is chosen to intentionally mismatch the true measurements \( g_{\text{true}, q}(p) \) so that the learning capabilities can be better observed. The profile is also Gaussian and follows \( \mathcal{N}(-60 - 2 \cdot x_q, 5) \), \( \forall q \), where

\[
\mathcal{N}(\mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left[ -\frac{(p - \mu)^2}{2\sigma^2} \right],
\]

defines the standard normal pdf in \( p \).

The profile still has physical plausibility, since the highest \( \mu \) is placed where the BS is. Figure 4.1 displays the initial model.

The control parameters are fixed at \( \kappa_{\text{max}} = 0.1 \), \( \phi = 3 \text{m} \), and \( \psi = 3 \text{dBm} \). This makes it possible to observe the convergence potential of SPLL without confounding of many effects. A faster and better convergence could be obtained with variable parameters but this was not the aim of this example.

A global RMS density error between the true model pdf that generated the measurements \( g_{\text{true}}(p) \) and the actual pdf \( g_k(p) \) recorded in the feature map at time \( k \) was
Figure 4.1: 1D simulation - initialisation

defined as:

\[
e_{\text{pdf},k} = \sqrt{\frac{1}{Q \cdot U} \sum_{q=1}^{Q} \sum_{u=1}^{U} \left( g_{k,q}(p_u) - g_{\text{true},q}(p_u) \right)^2},
\]

(4.15)

where \( U \) is the number of discrete field strength steps defined for the pdfs.

A mean position error is also defined using a fixed set of measurements as validation data. At each position \( x_q \), three measurement samples were drawn and used at each update iteration to evaluate the localisation accuracy. The error at time step \( k \) between the estimated position \( x_k(p_{M,q}) = x_{k,q} \) and the corresponding feature map position \( x_q \) that originated the measurement \( p_{M,q} \) is computed as:

\[
e_{\text{pos},k} = \frac{1}{Q} \sum_{q=1}^{Q} \sqrt{(x_{k,q} - x_q)^2},
\]

(4.16)

which is a particularisation of (3.37), when \( x \in \mathbb{R} \).

The experiments are run going from \( x_1 \) to \( x_{21} \) and back to \( x_1 \) 100 times, resulting in 4200 iterations. At each position \( x_q \), one measurement is randomly selected from the 1000 possible, and used for location estimation and feature map update according to the SPLL description.

**Exp. 4.1 (Unimodal Distribution).** For this setup, the true model \( g_{\text{true},q}(p) \) is defined
as:

$$g_{\text{true}, q} = \begin{cases} 
  \mathcal{N}(-20 - 2 \cdot x_q, 5), & \text{if } q < 11 \\
  \mathcal{N}(-40 - 2 \cdot x_q, 5), & \text{if } q \geq 11 
\end{cases}$$

which is shown in Fig. 4.2. The mismatch between the initialisation in Fig. 4.1 and the true model can be clearly observed. In this experiment, the discontinuity can be understood as being caused by a wall or by a door kept always shut, only opened to allow people to pass.

Figure 4.3 shows the learned pdf after 4200 iterations. Figures 4.4 and 4.5 show the evolution of $e_{\text{pdf}, k}$ and $e_{\text{pos}, k}$ respectively.

$e_{\text{pdf}, k}$ starts falling immediately and achieves its minimum after 1500 iterations. $e_{\text{pos}, k}$ falls accordingly and also reaches a minimum after 1500 iterations. It is notewor-
thy to compare Figs. 4.2 and 4.3 and verify that the learned pdf was brought by the SPLL to the position of the true model and that even the discontinuity was learned at the correct position.

**Exp. 4.2** (Bimodal Distribution). For this setup, the true model \( g_{\text{true},q}(p) \) is defined as:

\[
g_{\text{true},q} = \begin{cases} 
\mathcal{N}(-20 - 2 \cdot x_q, 5), & \text{if } q < 11 \\
0.5 \cdot \mathcal{N}(-20 - 2 \cdot x_q, 5) + 0.5 \cdot \mathcal{N}(-40 - 2 \cdot x_q, 5), & \text{if } q \geq 11
\end{cases}
\tag{4.18}
\]

which is shown in Fig. 4.6. The initialisation is the same as in the last experiment, displayed in Fig. 4.1, and again shows a clear mismatch with the true model. In this experiment, the discontinuity can be understood as caused by a door kept open half
of the time, for which no discontinuity is observed, and kept shut the other half, for which the discontinuity is the same as in the last experiment. Figure 4.7 shows the learned pdf after 4200 iterations. Figures 4.8 and 4.9 show the evolution of $e_{\text{pdf},k}$ and $e_{\text{pos},k}$, respectively.

This time, the SPLL fails to learn the true pmf properly, as a comparison between Figs. 4.6 and 4.7 reveals. Nevertheless the feature map is brought to a position that better describes the real distribution, rather than the initialisation, as can be observed by the error measures $e_{\text{pdf},k}$ and $e_{\text{pos},k}$. The $e_{\text{pdf},k}$ fall is not as steep as in the previous experiment, but the accuracy improvement showed by $e_{\text{pos},k}$ is still clearly present.
4.1.4.2 2D Test Bed

The 2D test bed was set to an area of $[50 \times 40]$m with 4 BSs placed at the corners of the defined area. The feature map is defined with 100 equally spaced positions and another 25 positions are reserved as validation points, which are used to estimate the actual feature map accuracy.

Figure 4.10 shows the simulation environment as well as the BSs, marked as black squares, the feature map positions, marked as gray crosses, and the validation positions, marked as gray triangles.

A trajectory was set in a way that all feature map positions are visited. The path starts at BS\textsubscript{1} and reaches BS\textsubscript{3} after passing through all positions within a specific sequence. Figure 4.11a shows the sequence used from BS\textsubscript{1} to BS\textsubscript{3}, and Fig. 4.11b the
way back from BS₃ to BS₁. Further repetitions are made using the same described path.

The density error $e_{\text{pdf},k}$ and the position error $e_{\text{pos},k}$ are evaluated at every time step $k$, according to (4.15) and (3.37). To evaluate $e_{\text{pos},k}$ a data set from the true model $G_{\text{true},q}$, with three measurements $\{p_{M1,q}, p_{M2,q}, p_{M3,q}\}$ at each position $x_q$, was reserved.

Since there are 4 BSs, the variables involved are described as vectors with dimension 4. Although the spatial dimension grew from 1 to 2, the dimension of the feature, that is, the pdf recorded in the feature map, is still 1. The difference is that for each position, there are now 4 distinct pdfs recorded.

In this way, for a feature map position $x_q$, the feature recorded is

$$G_q(p) = [g_{q,1}(p), g_{q,2}(p), g_{q,3}(p), g_{q,4}(p)]^T,$$

where each term $g_{q,n}(p)$ represents the pdf for the corresponding BS$_n$.

The initial model $G_{q,0}(p)$ at time $k = 0$ used in all experiments is based on unimodal
Figure 4.11: 2D simulation setup sequence (a) from BS$_1$ to BS$_3$ (b) from BS$_3$ to BS$_1$

Gaussian distributions, defined as:

$$G_{q,0}(p) = \begin{bmatrix} \mathcal{N}(-20 - 2 \cdot d_{q,1}, 5) \\ \mathcal{N}(-20 - 2 \cdot d_{q,2}, 5) \\ \mathcal{N}(-20 - 2 \cdot d_{q,3}, 5) \\ \mathcal{N}(-20 - 2 \cdot d_{q,4}, 5) \end{bmatrix},$$ (4.20)

with $d_{q,n}$ as the distance between $x_q$ and BS$_n$.

**Exp. 4.3 (Unimodal Distribution).** For this setup, the model $G_{\text{true}}(p)$ that originated the measurements was set as:

$$G_{\text{true},q} = \begin{cases} 
\mathcal{N}(-30 - 2 \cdot d_{q,1}, 3) & \text{if } w_{q,1} = 0 \\
\mathcal{N}(-50 - 2 \cdot d_{q,1}, 3) & \text{if } w_{q,1} = 1 \\
\mathcal{N}(-22 - 2 \cdot d_{q,2}, 3) & \text{if } w_{q,2} = 0 \\
\mathcal{N}(-42 - 2 \cdot d_{q,2}, 3) & \text{if } w_{q,2} = 1 \\
\mathcal{N}(-30 - 1 \cdot d_{q,3}, 3) & \text{if } w_{q,3} = 0 \\
\mathcal{N}(-50 - 1 \cdot d_{q,3}, 3) & \text{if } w_{q,3} = 1 \\
\mathcal{N}(-25 - 2 \cdot d_{q,4}, 3) & \text{if } w_{q,4} = 0 \\
\mathcal{N}(-45 - 2 \cdot d_{q,4}, 3) & \text{if } w_{q,4} = 1 
\end{cases},$$ (4.21)
where \( w_{q,n} \), for \( n = 1, 2, 3, 4 \), is a control flag that is 0 if position \( x_q \) is in LOS with BS\(_n\), and 1 otherwise.

The walking paths, as defined in Fig. 4.11 go directly through the wall. Since this is an illustrative example, it could be assumed that there are actually doors, which open only to permit a user to pass, but are otherwise kept (when the measurements are taken). In this way, a consistent drop in the RSS value should be observed in a real scenario. Only one mode is observed, either from one side or another of the wall.

The control parameters are fixed at \( \kappa_{\text{max}} = 0.1 \), \( \phi = 8\text{m} \), and \( \psi = 1\text{dBm} \). The walking path is used during 10000 iterations, always selecting random measurements. 

\( e_{\text{pdf},k} \) and \( e_{\text{pos},k} \) are displayed in Figs. 4.12 and 4.13. The individual accuracy at each of the validation points is presented in Fig. 4.14.

From Fig. 4.12 it can be seen that \( e_{\text{pdf},k} \) starts falling right after the start and it
reaches a noisy but steady state after 1000 iterations. $e_{pos,k}$ starts at almost 9m and despite the noisy behaviour of $e_{pdf,k}$, it also decays to 4m but much more slowly than $e_{pdf,k}$. An improvement of 41.93% was achieved in $e_{pdf,k}$ and of 53.23% in $e_{pos,k}$. The results and experiment setup are summarised later in Table 4.2.

Figure 4.14 shows the spatial distribution of the localisation error at $k = 10000$. It can be seen that the validation positions near the wall are the most penalised. This can be explained by the constant adaptation radius imposed by this experiment. Every time a position on the other side of the wall is updated, other positions on the side of the validation point are also updated, but in the wrong direction. This suggests that the control parameters should be made smaller with increasing time after the self-organisation is achieved (refer to section 3.3).

Exp. 4.4 (Bimodal Distribution). The actual true model $G_{true,q}$ which generated the
measurement data is defined as:

\[
G_{\text{true},q} = \begin{cases} 
\mathcal{N}(-20 - 2 \cdot d_{q,1}, 5) & \text{if } w_{q,1} = 0 \\
0.5\mathcal{N}(-20 - 2 \cdot d_{q,1}, 5) + 0.5\mathcal{N}(-40 - 2 \cdot d_{q,1}, 5) & \text{if } w_{q,1} = 1 \\
\mathcal{N}(-20 - 2 \cdot d_{q,2}, 5) & \text{if } w_{q,2} = 0 \\
0.5\mathcal{N}(-20 - 2 \cdot d_{q,2}, 5) + 0.5\mathcal{N}(-40 - 2 \cdot d_{q,2}, 5) & \text{if } w_{q,2} = 1 \\
\mathcal{N}(-20 - 2 \cdot d_{q,3}, 5) & \text{if } w_{q,3} = 0 \\
0.5\mathcal{N}(-20 - 2 \cdot d_{q,3}, 5) + 0.5\mathcal{N}(-40 - 2 \cdot d_{q,3}, 5) & \text{if } w_{q,3} = 1 \\
\mathcal{N}(-20 - 2 \cdot d_{q,4}, 5) & \text{if } w_{q,4} = 0 \\
0.5\mathcal{N}(-20 - 2 \cdot d_{q,4}, 5) + 0.5\mathcal{N}(-40 - 2 \cdot d_{q,4}, 5) & \text{if } w_{q,4} = 1 
\end{cases} 
\]

(4.22)

where, as defined in the previous setup, \(w_{q,n}\) with \(n = 1, 2, 3, 4\), is a control flag that is 0 if position \(\mathbf{x}_q\) is in LOS with BS\(_n\), and 1 otherwise.

The walking path, as defined in Fig. 4.11, goes again directly through the wall. It is assumed again that there are doors, but this time either as a wide roll up door, often used in garages, or as a series of doors side by side as often found in theatres or convention centres. The bimodality described by (4.22) indicates that such doors were open for half of the measurements, for which there was no drop in the RSS value, and closed during the other half, dropping the RSS value as in the previous setup.

The control parameters are again fixed at \(\kappa_{\text{max}} = 0.1\), \(\phi = 8\)m, and \(\psi = 1\)dBm. The same walking path from Exp. 4.3 is used during 10000 iterations, always selecting random measurements.

\(e_{pdf,k}\) and \(e_{pos,k}\) are displayed in Figs. 4.15 and 4.16. The individual accuracy at each of the validation points is presented in Fig. 4.17.

From Fig. 4.15 it can be seen that, even with this more complex distribution profile, \(e_{pdf,k}\) falls again right after the start and it reaches a noisy but steady state after 1000 iterations. \(e_{pos,k}\) starts at almost 11m and despite the noisy behaviour of \(e_{pdf,k}\), it also decays to 4.5m but much more slowly than \(e_{pdf,k}\). However, the behaviour of \(e_{pos,k}\) oscillates much more than in Exp. 4.3. An improvement of 60.4% was achieved in \(e_{pdf,k}\) and of 57.16% in \(e_{pos,k}\). The results and experiment setup are summarised later in Table 4.2.

Figure 4.17 shows the spatial distribution of the localisation error at \(k = 10000\).
Figure 4.15: 2D bimodal simulation - pdf error

Figure 4.16: 2D bimodal simulation - position error

Figure 4.17: 2D bimodal simulation - position error at $k = 10000$
Figure 4.18: 2D bimodal simulation - pdf at $k = 10000$, $q = 96$, and $n = 3$

It can be seen that the validation positions near the wall are still the most penalised. Although $e_{pdf,k}$ decreased with more iterations, the final state achieved for the pdfs cannot be observed with the shown plots. For the 1D setup it was possible to display the results for all pdfs in one single plot. For the 2D case, there are 100 distinct positions, each with 4 individual pdfs (one for each BS). Figure 4.18 synthesises the main aspects of learning, showing the true and the learned pdf at position $x_{96}$, for BS$_3$ and for $k = 10000$. From this plot, it is clear that the SPLL in this case failed to represent the two modes correctly, as was the case in Exp. 4.2.

4.2 Extended SPLL

Experiments 4.2 and 4.4 show that the initial formulation of SPLL is not adequate to learn complex distributions, or complex scenarios when not enough information is available. Therefore, some improvement of the original algorithm is needed. For both SLL and SPLL there are two basic ways to improve the algorithm: either improve the localisation accuracy for the same input information, i.e., how to achieve better location estimates with the same feature map and the same measurement, or improve the learn rule, i.e., how to use the same input information to update the feature map in a better way.

This section deals with two realistic assumptions that can be used advantageously to improve SPLL: multimeasurement and tracking.
4.2.1 Multimeasurement

If measurements are gathered fast enough, so that in one position more than one RSS measurement is available at once, then this can be used for both accuracy and learning improvement.

In the presence of more than one measurement for one single location, a simple approach would be to average the available measurements and make only one location query with this averaged input. As SPLL uses probabilistic localisation, averaging is not a good solution, since in this way relevant probabilistic information is lost through averaging.

Localisation In Chapter 2, equation (2.6) showed the core of probabilistic localisation with Bayes rule, using one single measurement $p_M$, and the practical application of it in subsection 4.1.3, equation (4.11). For the case where $V$ measurements \{$p_{M,1}, p_{M,2}, \ldots, p_{M,V}$\} are available, (4.11) is modified into:

\[
\Pr(x_q|p_{M,1}, \ldots, p_{M,V}) = \frac{\Pr(p_{M,1}, \ldots, p_{M,V}|x_q) \cdot Pr(x_q)}{\sum_{i=1}^{Q} \Pr(p_{M,1}, \ldots, p_{M,V}|x_i) \cdot Pr(x_i)} \tag{4.23}
\]

Under the assumption that the $V$ measurements are independent, then:

\[
\Pr(p_{M,1}, \ldots, p_{M,V}|x_q) = \Pr(p_{M,1}|x_q) \cdot \ldots \cdot \Pr(p_{M,V}|x_q) = \prod_{v=1}^{V} \Pr(p_{M,v}|x_q), \tag{4.24}
\]

and further assuming, as in (2.7), that the measurements from $N$ BSs are independent, then:

\[
\Pr(p_{M,v}|x_q) = \prod_{n=1}^{N} \Pr(p_{M,v,n}|x_q) \tag{4.25}
\]

Due to numeric precision by computer implementation, it is convenient to normalise (4.25) before applying it into (4.24), otherwise this product can turn to be zero. This can happen, for example, if there are many weak BSs received at a particular position.
Therefore, (4.24) becomes:

\[
\Pr(p_{M,1}, \cdots, p_{M,V}|x) = \prod_{v=1}^{V} \frac{\prod_{n=1}^{N} \Pr(p_{M,v,n}|x)}{\sum_{i=1}^{Q} \prod_{n=1}^{N} \Pr(p_{M,v,n}|x_i)},
\]

(4.26)

with the sum on \(i\) in the denominator as normalising factor.

Again, if there is no a priori information about where is the actual user location, then \(\Pr(x_q) = \frac{1}{Q}\), meaning that all positions on the feature map are equiprobable.

Similarly to (2.11), the location estimate is then retrieved using MMSE:

\[
x = E[x_M|p_{M,1}, \cdots, p_{M,V}] = \sum_{q=1}^{Q} x_q \cdot \Pr(x_q|p_{M,1}, \cdots, p_{M,V})
\]

(4.27)

**Learning** The SPLL learning, based on non-parametric density estimation as in (4.6), can be improved with multimeasurements as well. The kernel function \(f_k\) in (4.7) is modified into:

\[
f_k(p) = \frac{\kappa}{V} \cdot \sum_{v=1}^{V} \exp \left( - \left( \frac{p - p_{M,k,v}}{\psi} \right)^2 \right),
\]

(4.28)

so that the \(V\) measurements can be used at once at the same time step \(k\) for learning. During this time step (4.28) uses all \(V\) RSS values to estimate a pdf according to the classical kernel method, without a forgetting factor, as in (4.2), and weighted by \(\kappa\). The composite kernel for \(f_k\) is then applied to (4.6) like the single measurement SPLL does.

**4.2.2 Tracking**

\(\Pr(x_q)\) is usually set to a constant, assuming it is uniformly distributed, if there is no prior information about where a user is located. However, if the time elapsed between two consecutive location queries is not big, then the user’s previously estimated position \(x_{k-1}\) at discrete time \(k - 1\) can be used to shape \(\Pr(x_q)\) accordingly. In this way a simple form of tracking is achieved.
Pr(\(x_q\)) then becomes the conditional probability Pr(\(x_{k,q}|x_{k-1}\)), which represents how likely a user is located at \(x_{k,q}\) now, given that in the previous location estimate he was at \(x_{k-1}\). A simple approach to realistically model this dependency is to use a movement model.

Assuming that the user remained in the same position \(x_{k-1}\) since the last location query, a pdf can be built placing its peak at \(x_{k-1}\). If the continuous time difference \(\tau\) between consecutive location queries is big, then a measure for the uncertainty should be brought to the pdf. Considering the maximum speed \(\nu_{\text{max}}\) a user can achieve then a circle with radius \(R = \nu_{\text{max}} \cdot \tau\) can be defined around \(x_{k-1}\) that is equiprobable in respect to Pr(\(x_q\)). The following pdf \(g(x|x_{k-1})\) encapsulates these considerations:

\[
g(x|x_{k-1}) = \begin{cases} 
  h \cdot \exp \left( -\frac{(r_{k-1} - R)^2}{2\sigma^2} \right) & \text{if } r_{k-1} \geq R \\
  h & \text{if } r_{k-1} < R 
\end{cases}
\]

with

\[
r_{k-1} = \sqrt{(x - x_{k-1})^T(x - x_{k-1})}
\]

For 1D feature maps \(h\) must be:

\[
h = h_{1D} = \frac{1}{2R + \sigma\sqrt{2\pi}}
\]

and for 2D feature maps:

\[
h = h_{2D} = \frac{1}{2\pi \cdot \sigma^2 + R \cdot \sigma\sqrt{2\pi^3} + \pi \cdot R^2}
\]

which ensures that

\[
\int_{-\infty}^{+\infty} g(x|x_{k-1}) \, dx = 1
\]

Proofs of (4.33) for both the 1D and the 2D case are presented in appendix B.

Figure 4.19 shows qualitative examples in 1D and 2D for (4.29) with different \(R_s\) for the same \(\sigma\). \(R_1 = 0\)m for \(\tau_1 = 0\)s and in this case, (4.29) reduces to the known Gaussian distribution, with mean \(x_{k-1}\) and standard deviation \(\sigma\). \(R\) and \(\sigma\) are referred to as tracking parameters.
4.2.3 Simulated Examples

Experiments 4.2 and 4.4, which motivated the improvements introduced to the SPLL in this section, are repeated here with the modified algorithm.

4.2.3.1 1D Test Bed

Exp. 4.5 (Bimodal Distribution). In Sec. 4.1.4.1 the SPLL failed to learn the bimodal distribution shown in Fig. 4.6. The experiment is again initialised with the mismatched model displayed in Fig. 4.1.

The control parameters remain the same as those used in the previous 1D simulation: \( \kappa_{\text{max}} = 0.1, \phi = 3\text{m} \) and \( \psi = 3\text{dBm} \).

This time the multimeasurement is considered with \( V = 5 \) for each time step \( k \). The localisation is then accomplished according to (4.27). For the calculation of \( \Pr(x_q) \) the conditional density defined in (4.29) is used, with \( R = 4\text{m} \) and \( \sigma = 2\text{m} \).

Figure 4.20 shows the learned pdf after 4200 iterations. Figures 4.21 and 4.22 show the evolution of \( e_{\text{pdf},k} \) and \( e_{\text{pos},k} \), respectively.

This time the SPLL learns the true pmf properly, as a comparison between Figs. 4.6 and 4.20 shows. The feature map is brought once again to a position that better describes the real distribution, rather than the initialisation, as can be observed by the error measures \( e_{\text{pdf},k} \) and \( e_{\text{pos},k} \). The \( e_{\text{pdf},k} \) fall is even steeper than with the unimodal simulation, which can be explained by the greater amount of information brought by...
Figure 4.20: 1D bimodal simulation - learned pdf at $k = 4200$

Figure 4.21: 1D bimodal simulation - pdf error

Figure 4.22: 1D bimodal simulation - position error
the multimeasurement and the tracking. $e_{\text{pos},k}$ falls accordingly, reaching the same accuracy as in the unimodal simulation.

The Table 4.1 summarises the results from Exps. 4.1, 4.2, and 4.5.

<table>
<thead>
<tr>
<th>Exp.</th>
<th>4.1</th>
<th>4.2</th>
<th>4.5</th>
<th>4.1</th>
<th>4.2</th>
<th>4.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length (m)</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>FM Positions</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>$\rho_{\text{pos}}$ (m)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\kappa_{\text{max}}$</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>$\phi$ (m)</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$\psi$ (dBm)</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$R$ (m)</td>
<td>--</td>
<td>4</td>
<td>--</td>
<td>4</td>
<td>--</td>
<td>4</td>
</tr>
<tr>
<td>$\sigma$ (m)</td>
<td>--</td>
<td>2</td>
<td>--</td>
<td>2</td>
<td>--</td>
<td>2</td>
</tr>
<tr>
<td>$V$</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>$k$</td>
<td>0</td>
<td>7.34</td>
<td>8.74</td>
<td>8.74</td>
<td>0.033</td>
<td>0.031</td>
</tr>
<tr>
<td>$e_{\text{pos}}$ (m)</td>
<td>0.94</td>
<td>1.68</td>
<td>1.33</td>
<td>0.011</td>
<td>0.019</td>
<td>0.006</td>
</tr>
<tr>
<td>$e_{\text{pdf}}$ (dBm$^{-1}$)</td>
<td>87.26</td>
<td>80.80</td>
<td>84.78</td>
<td>67.19</td>
<td>39.66</td>
<td>79.21</td>
</tr>
</tbody>
</table>

Table 4.1: SPLL Simulated 1D Experiments

4.2.3.2 2D Test Bed

Exp. 4.6 (Bimodal Distribution). In Sec. 4.1.4.2 the SPLL failed to learn the bimodal distribution shown in Fig. 4.18. The experiment is again initialised with the mismatched model given in (4.20).

The control parameters remain the same as those used previously: $\kappa_{\text{max}} = 0.1$, $\phi = 8$ m, and $\psi = 1$ dBm. The same walking path from Fig. 4.11 is used. Additionally, the multimeasurement is considered with $V = 15$ for each time step $k$. The localisation is then accomplished according to (4.27). For the calculation of $\Pr(x_q)$ the conditional density defined in (4.29) is used, with $R = 5$ m and $\sigma = 1$ m. Figures 4.23 and 4.24 show the evolution of $e_{\text{pdf},k}$ and $e_{\text{pos},k}$, respectively. Comparing these with Figs. 4.15 and 4.16 it is clear that the Extended SPLL goes further in the feature map improvement. The fall of $e_{\text{pdf},k}$ is even more pronounced. However, the final accuracy $e_{\text{pos},k}$ was not noticeably improved, but it converged faster and with smaller oscillations.

Figure 4.25 shows the spatial distribution of the localisation error at $k = 10000$. Comparing it with Fig. 4.17 it can be seen that the validation positions near the wall
Figure 4.23: 2D bimodal simulation - pdf error

Figure 4.24: 2D bimodal simulation - position error

Figure 4.25: 2D bimodal simulation - position error at $k = 10000$
are now less penalised. This is made clear in Fig. 4.26, representing the same position

![Figure 4.26: 2D bimodal simulation - pdf at $k = 10000$, $q = 96$, and $n = 3$](image)

and BS as in Fig. 4.18. This time the two modes are correctly learned, showing the improvement achieved with the Extended SPLL.

The Table 4.2 summarises the results from Exps. 4.3, 4.4, and 4.6.

<table>
<thead>
<tr>
<th>Exp.</th>
<th>4.3</th>
<th>4.4</th>
<th>4.6</th>
<th>4.3</th>
<th>4.4</th>
<th>4.6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area (m²)</td>
<td></td>
<td></td>
<td>2000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FM Positions</td>
<td></td>
<td></td>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p_{pos}$ (m)</td>
<td></td>
<td></td>
<td>4.97</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$k_{max}$</td>
<td></td>
<td></td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\phi$ (m)</td>
<td></td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\psi$ (dBm)</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R$ (m)</td>
<td>—</td>
<td>5</td>
<td>—</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sigma$ (m)</td>
<td>—</td>
<td>1</td>
<td>—</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$V$</td>
<td>1</td>
<td>15</td>
<td>1</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$k_{pos}$ (m)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$e_{pdf}$ (dBm⁻¹)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>error</td>
<td>0</td>
<td>8.95</td>
<td>10.71</td>
<td>10.71</td>
<td>0.0015</td>
<td>0.0022</td>
</tr>
<tr>
<td>4200</td>
<td>4.18</td>
<td>4.59</td>
<td>4.26</td>
<td>0.00089</td>
<td>0.00089</td>
<td>0.00053</td>
</tr>
<tr>
<td>improvement (%)</td>
<td>53.25</td>
<td>57.16</td>
<td>60.19</td>
<td>41.93</td>
<td>60.40</td>
<td>76.43</td>
</tr>
</tbody>
</table>

Table 4.2: SPLL Simulated 2D Experiments

### 4.3 SPLL: Real World Experiments

**Exp. 4.7** (1D Office Corridor). For this experiment, WLAN measurements were taken in a corridor with a length of 31.2m, with a BS at $x_1 = 0m$ and steps of 1.2m between
Figure 4.27: 1D Real world experiment: true model

the feature map positions $x_q = (q - 1) \cdot 1.2$ with $q = \{1, \cdots, 27\}$. Each position was measured 200 times and the corridor has a thick metal door at $x_{14} = 15.6$m, which was kept shut during half of the time, and fully open during the other half.

The feature map was initialised using $\mathcal{N}(-60 - x_q, 5), \forall q$. Since there is no a priori knowledge of the true model, a representation of its pdf is achieved using common non-parametric density estimation, but only for comparison with the learned feature map. The mismatch between the measurements and the initial model are presented in Figs. 4.27 and 4.28, respectively. Noticeable are the discontinuity due to the door and the bimodal distribution that appears after the door, representing both states: open and closed. Also remarkable is the difference in the wideness of the true and the initial feature map pmfs.

The experiment was run going from $x_1$ to $x_{27}$ and back to $x_1$ 150 times, resulting in 8100 iterations. At each position $x_q$, five measurements were randomly selected from the 200 possible. The control parameters were set to $\kappa_{\text{max}} = 0.1$, $\phi = 3$m, $\psi = 3$dBm and the tracking parameters $R = 4$m and $\sigma = 2$m, which are the same values as used in the SPLL 1D simulations. Figures 4.29 and 4.30 show the evolution of $\epsilon_{\text{pdf},k}$ and $\epsilon_{\text{pos},k}$ respectively.

$\epsilon_{\text{pdf},k}$ starts falling after a short rise and achieves its minimum after 2000 iterations. $\epsilon_{\text{pos},k}$ falls accordingly, showing that the matching in the model is directly related to achieved accuracy. It takes somewhat longer to achieve the minimum than in the
Figure 4.28: 1D Real world experiment: initialisation

Figure 4.29: 1D Real world experiment: pdf error

Figure 4.30: 1D Real world experiment: position error
simulated example, also the minimum error itself is bigger. This is because the measurement profile here is more complex, and the pdfs used for comparison are only a representation of the unknown true pdfs. Nevertheless, the feature map goes from its initially false start state to a much better representation of the measurements as Fig. 4.31 shows. In this plot the learned feature map at the 8100th iteration is displayed.

Once again it is possible to verify that the learned pdf was brought by the SPLL to the region of the true model and that also the discontinuity and the bimodal distribution after the door were learned at the right positions $x_q$, as Figs. 4.27 and 4.31 show.

Exp. 4.8 (2D Office Environment). This experiment was performed in an office environment with size $[15 \times 33]$m. Figure 4.32a shows the plant with the spatial division imposed by the building. The figure also displays the 119 training positions, marked as crosses, and the 21 validation positions, marked as triangles. The area is covered by 8 WLAN BSs, marked as squared dots.

The training and validation sets were gathered by measuring each position 200 times, in the morning, during lunch time and in the afternoon, totaling 600 measurements with distinct time profiles. Nevertheless, the 600 measurements were considered together disregarding the conditions with which they were taken.

A walking path was defined passing by all training positions and returning to the start point. The sequence of positions is displayed in Fig. 4.32b. Further repetitions
The feature map was initialised using $\mathcal{N}(-20 - 2 \cdot d_{q,n}, 10)$, where $d_{q,n}$ is the distance from the feature map position $x_q$ to BS$_n$.

The experiment was run following the path in Fig. 4.32b for 10000 iterations. The number $V$ of measurements per time step $k$ was set to 5, which were randomly selected from the 600 stored values, at each position $x_q$. The control parameters were set to $\kappa_{\text{max}} = 0.1$, $\phi = 5\text{m}$, $\psi = 1\text{dBm}$ and the tracking parameters $R = 1\text{m}$ and $\sigma = 1\text{m}$. To evaluate $e_{\text{pos},k}$, five measurements were considered at each validation position. Figures 4.33 and 4.34 show the evolution of $e_{\text{pdf},k}$ and $e_{\text{pos},k}$, respectively. From Fig. 4.33 it can be seen that $e_{\text{pdf}}$ drops and stabilises before 1000 iterations, presenting an oscillatory pattern that corresponds to the repetition of the followed path. The $e_{\text{pos}}$ also drops right after the start and with a stronger oscillatory effect. However, two successive
drops, after $k = 6000$ and at $k = 8000$ indicate that new self-organising states were reached, that better represent the learn patterns. The control parameters were kept constant all the time, but the oscillatory behaviour decreased noticeably at these new states, showing their stability. The final spatial accuracy at $k = 10000$ can be seen in Fig. 4.35. It is noteworthy that almost all elements of the validation set were located inside the correct room, or corridor.

As an example of the capability of SPLL to learn complex composite distributions, Fig. 4.36 shows the true and learnt pdf at the 10000th iteration, at position $x_{11}$ for BS7.

The Table 4.3 summarises the results from Exps. 4.7 and 4.8.

<table>
<thead>
<tr>
<th>Exp.</th>
<th>4.7 (1D)</th>
<th>4.8 (2D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>31.2m</td>
<td>480m²</td>
</tr>
<tr>
<td>FM Positions</td>
<td>27</td>
<td>119</td>
</tr>
<tr>
<td>$\rho_{pos}$ (m)</td>
<td>1.2</td>
<td>2.21</td>
</tr>
<tr>
<td>$\kappa_{max}$</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>$\phi$ (m)</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>$\psi$ (dBm)</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>$R$ (m)</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>$\sigma$ (m)</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$V$</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>error</th>
<th>$k$</th>
<th>$e_{pos}$ (m)</th>
<th>$e_{pdf}$ (dBm$^{-1}$)</th>
<th>$k$</th>
<th>$e_{pos}$ (m)</th>
<th>$e_{pdf}$ (dBm$^{-1}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>11.76</td>
<td>0.039</td>
<td>0</td>
<td>13.48</td>
<td>0.018</td>
</tr>
<tr>
<td></td>
<td>8100</td>
<td>2.63</td>
<td>0.019</td>
<td>10000</td>
<td>2.89</td>
<td>0.0044</td>
</tr>
</tbody>
</table>

Table 4.3: SPLL Real World Experiment Results
Figure 4.34: 2D Real world experiment: position error

Figure 4.35: 2D Real world experiment: position error at $k = 10000$
Figure 4.36: 2D Real world experiment: pdf at $k = 10000$, $q = 11$, and $n = 7$
Chapter 5

Conclusions

Indoor radio positioning systems are usually associated with high costs. They either use proprietary sensors or tags, which give accuracies of some metres or even fractions of it, or they use off-the-shelf equipment, which is cheaper, but require extensive calibration procedures in order to achieve a desired accuracy.

This thesis focuses on indoor location systems which use RSS as the measured feature to retrieve a user location. The calibration phase for such systems, which must be performed before taking them into operation, is a cumbersome and time consuming task that contributes a lot to implementation and maintenance costs. The data retrieved in a normal calibration scenario is regarded as labelled data, that is, extra information, such as location and time, for example, must be stored in the database together with the measurements.

The work presented here introduces two algorithms that automatically learn and store measurements in order to eliminate the calibration phase. The data is fetched disregarding the exact location where it was taken. In this sense, the data is regarded as unlabelled. In this way a system can be initialised with a rough model, respecting at least physical plausibilities, and used to locate users right after start. Through successive location queries, the system is fed with new information that is used to improve the model using learning algorithms.

The Simultaneous Localisation and Learning (SLL) was the first algorithm implemented and used the mean value of RSS measurements as feature for the stored database, also called radio map. The learning rule can be seen as a particular case of Koho-
nen SOMs, and to some extent as a step further in their application and mathematical interpretation (SOMs are defined over a discrete grid of neurons while the SLL does not need to be). Mathematical proofs and statistical conditions were presented, showing the learn capabilities of the algorithm and also validated using theoretical simulation scenarios, which highlighted particular properties of the algorithm, and using real test scenarios, with installed infrastructure based in DECT and WLAN networks.

The Simultaneous Probabilistic Localisation and Learning (SPLL) was the natural next step of the SLL. While the SLL dealt with the mean value of the RSS, the SPLL used the whole probability distribution. In order to learn and use the pdfs of the RSS, the SLL algorithm was modified and merged with non parametric density estimation techniques. Again, mathematical proofs were shown, together with simulation and real case scenarios which validate the algorithm.

At first glance, both algorithms seem to have the same issues as the SOMs, and as many neural networks have: the tuning of the learn parameters are strongly dependent on heuristics and sometimes on trial and error, which can be inefficient when dealing with real cases. This is not the case with SLL and SPLL, as most of the control and learn parameters have physical interpretations that permit the range of those parameters to be constrained. For example, the spatial adaptation width $\phi$ should have a magnitude compatible with the density of feature map positions. The tracking radius, given by $R$ and $\sigma$, implicitly includes the maximum speed a user is expected to move, and the elapsed time since the last measurement. The adaptation width in power $\psi$ should reflect the discretisation interval used to represent the pdfs as pmfs. The learning rate $\kappa$ reflects the amount of trust in the calculated update based on actual measurements. This is the only one of the parameters that has no direct physical interpretation, but still can be set according to the subjective desired behaviour of the algorithm. For example, right after the system start, as the feature map is very rough and unprecise, $\kappa$ should be big so that the map is rearranged to another self-organising state more quickly. Then it should be made smaller so that fine details are also learned and also as a way to freeze the learning if the desired accuracy is achieved. Another influence of $\kappa$ is to control how fast new measurements replace the old ones stored in the feature map. This forgetting effect, imposed by the learn algorithm, should not be faster than the natural changes due to multimodes, otherwise there is a risk of only
learning single modes when considering SPLL.

The SPLL logically requires a heavier load on memory and processing power, as algorithms dealing with whole statistical distributions are more complex than those dealing only with mean values, as the SLL does. However, details of multimodal behaviour, often found in indoor environments, can only be learnt by the SPLL. The use of mean values for indoor systems often underestimates the environment complexity, especially when dealing with doors and windows, which easily impose multimodal behaviour to the radio propagation.

The start model can be made as complex as desired, but this has to be considered against the advantage of SLL and SPLL. Since the algorithms are going to adapt the map anyway, too much time should not be invested on the initial model. It must only provide physically plausible conditions so that a rough accuracy can be achieved at the beginning. This condition of plausibility cannot be neglected, or the location queries will also deliver implausible results, leading to algorithm divergence.

Another critical aspect, often found within the neural networks scope, is the overtraining. With SLL and SPLL, this means that a user remains still in a position and the feature map is continuously updated, leading to a flattening of the feature map, as shown in the mathematical proofs. From the statistical conditions, it is stated that there must be measurements available from everywhere on the map. If for some reason, a particular area is more often visited than others (such as a corridor), or there are repeated measurements at a same single position, there should be a way to identify these cases and to adjust the learn parameters so that an overtraining condition is avoided.

One aspect of both algorithms not explored in this work, but intrinsic to them and of great advantage when compared with other works in this area, is the dynamic adaptation to propagation changes. With the usual systems, every time some change in the propagation profile is imposed, such as the installation of new or displacement of old BSs, failure of existent BSs, or geometrical changes in space (as often found in office environments, where walls can easily be moved), a recalibration is again required in order to keep the location system operational. With SLL and SPLL, there is no need to stop the system since the algorithm is capable of adapting to such changes. A reset of the control parameters might be required, making $\phi$ and $\kappa$ bigger again for example,
so that the algorithm can move from a stable self-organising state to another state that better represents these changes. Here again, the speed of the forgetting effect, directly linked to $\kappa$, also dictates the speed with which this new state will replace the old one. In this way, not only the implementation costs are significantly reduced but also the maintenance costs of the system.
## Appendix A

### Nomenclature

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>area</td>
<td>112</td>
</tr>
<tr>
<td>$A_j$</td>
<td>auxiliary variable, representing the denominator of (4.6)</td>
<td>65</td>
</tr>
<tr>
<td>$B$</td>
<td>propagation law</td>
<td>38</td>
</tr>
<tr>
<td>$\text{BS}_n$</td>
<td>$n^{th}$ base station</td>
<td>14</td>
</tr>
<tr>
<td>$c$</td>
<td>speed of light</td>
<td>12</td>
</tr>
<tr>
<td>$c$</td>
<td>subscript index referring to centre neuron or position</td>
<td>25</td>
</tr>
<tr>
<td>$d_{\text{ref}}$</td>
<td>reference distance</td>
<td>8</td>
</tr>
<tr>
<td>$d$</td>
<td>general distance</td>
<td>8</td>
</tr>
<tr>
<td>$d_{ij}$</td>
<td>distance between $x_i$ and $x_j$</td>
<td>9</td>
</tr>
<tr>
<td>$d_{ci}$</td>
<td>distance from neuron $c$ to neuron $i$</td>
<td>25</td>
</tr>
<tr>
<td>$d_{c,k}$</td>
<td>general distance from any position $x$ to $x_{c,k}$</td>
<td>57</td>
</tr>
<tr>
<td>$d_{q,n}$</td>
<td>distance from position $x_q$ to $\text{BS}_n$</td>
<td>67</td>
</tr>
<tr>
<td>$d_q$</td>
<td>general distance from position $x_q$ to $x$</td>
<td>69</td>
</tr>
<tr>
<td>$D(d_n, \gamma_n)$</td>
<td>monotonic propagation function for $\text{BS}_n$</td>
<td>27</td>
</tr>
<tr>
<td>$e_{\text{FM},k}$</td>
<td>feature map error at time $k$</td>
<td>44</td>
</tr>
<tr>
<td>$e_{\text{pos},k}$</td>
<td>position or localisation error at time $k$</td>
<td>56</td>
</tr>
<tr>
<td>$e_{\text{pdf},k}$</td>
<td>pdf error at time $k$</td>
<td>71</td>
</tr>
</tbody>
</table>

*Continued on next page*
\( f_{c,k} \) SLL weighting function at time \( k \)
\( f_i \) kernel function
\( F_k \) SLL utility function
\( g(p, x) \) pdf
\( g(p_n|x_q) \) pdf of \( p_n \) given \( x_q \)
\( g_k(p) \) pdf at time \( k \)
\( g_{\text{true},q} \) true pdf that originated the measurements in SPLL
\( G_q \) composite pdf set at \( x_q \)
\( G_{\text{true}} \) composite true pdf
\( h \) height
\( h \) kernel method smoothing parameter
\( h_{ci} \) SOM neighbourhood function
\( i \) generic subscript index
\( j \) generic subscript index
\( k \) number of neighbours in the kNN method
\( k \) discrete time index
\( K \) kernel function
\( L \) 1D map length
\( L_x \) 2D map boundary in \( x \) axis
\( L_y \) 2D map boundary in \( y \) axis
\( m_i \) SOM neuron \( i \) weight vector
\( M_{Z,k} \) SLL utility function
\( n \) index for the number of BSs
\( N \) total number of BSs
\( N \) normal probability density function
\( p_r \) received power
\( p_{\text{ref}} \) received power at \( d_{\text{ref}} \)
\( p_q \) received power at \( x_q \)

Continued on next page
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_{M,n} )</td>
<td>measured power from BS(_n)</td>
</tr>
<tr>
<td>( p_{q,n} )</td>
<td>received power from BS(_n) at ( x_q )</td>
</tr>
<tr>
<td>( p_n )</td>
<td>power from BS(_n)</td>
</tr>
<tr>
<td>( p_{out} )</td>
<td>output power from a transmitting device</td>
</tr>
<tr>
<td>( p_k )</td>
<td>SLL radio map at time ( k )</td>
</tr>
<tr>
<td>( p_{n,k} )</td>
<td>received power from BS(_n) at time ( k )</td>
</tr>
<tr>
<td>( p_{M+} )</td>
<td>upper boundary of limit area</td>
</tr>
<tr>
<td>( p_{M-} )</td>
<td>lower boundary of limit area</td>
</tr>
<tr>
<td>( p_{\text{true}} )</td>
<td>true radio map model vector</td>
</tr>
<tr>
<td>( p_{\text{true},q} )</td>
<td>true radio map model at ( x_q )</td>
</tr>
<tr>
<td>( p_{\text{true},q,n} )</td>
<td>true radio map model from BS(_n) at ( x_q )</td>
</tr>
<tr>
<td>( p_{\text{training},q,n} )</td>
<td>training data from BS(_n) at ( x_q )</td>
</tr>
<tr>
<td>( p_{\text{validation},q} )</td>
<td>validation measurements at ( x_q )</td>
</tr>
<tr>
<td>( P_{M,k} )</td>
<td>SLL utility function</td>
</tr>
<tr>
<td>( q )</td>
<td>subscript index for the number of discrete positions in a feature map</td>
</tr>
<tr>
<td>( Q )</td>
<td>total number of discrete positions in a feature map</td>
</tr>
<tr>
<td>( r_i )</td>
<td>radius of circle centred at ( x_i )</td>
</tr>
<tr>
<td>( R )</td>
<td>radius</td>
</tr>
<tr>
<td>( S_{i,2,k} )</td>
<td>SLL utility function</td>
</tr>
<tr>
<td>( S_i )</td>
<td>area of ( f_i )</td>
</tr>
<tr>
<td>( U )</td>
<td>total number of discrete field strength points for a pdf or pmf</td>
</tr>
<tr>
<td>( u )</td>
<td>subscript index for the number of discrete field strength points for a pdf or pmf</td>
</tr>
<tr>
<td>( V )</td>
<td>total number of measurements considered at one iteration</td>
</tr>
<tr>
<td>( V )</td>
<td>Volume</td>
</tr>
<tr>
<td>( v )</td>
<td>subscript index for the number of measurements considered at one iteration</td>
</tr>
</tbody>
</table>

*Continued on next page*
$W$ total number of observations in the kernel method 62
$x$ position in $x$ coordinate 9
$x_q$ $q^{th}$ discrete position in a feature map 14
$x_{M}$ position where a measurement $p_M$ was taken 14
$x_{c,k}$ SLL estimated centre position at time $k$ 29
$x_f$ fixed position 33
$x_{M,1:k}$ set of measurement positions from start until time $k$ 36
$x_{c,1:k}$ set of centre positions from start until time $k$ 36
$y$ position in $y$ coordinate 9
$z$ kernel function auxiliary variable 62
$Z_k$ SLL utility function 36
$\alpha$ SOM learning rate 25
$\gamma$ model attenuation factor 8
$\Delta p_{out}$ variation or offset on $p_{out}$ 41
$\Delta \gamma$ variation or offset on $\gamma$ 41
$\Delta x_M$ variation or offset on $x_M$ 44
$\zeta$ noise term 35
$\theta$ generic angle 9
$\kappa$ learning rate, or adaptation amplitude 29
$\kappa_{\text{max}}$ maximum learning rate for SPLL 69
$\phi$ adaptation width in space for SLL and SPLL 29
$\mu$ mean value 35
$\nu_{\text{max}}$ maximum speed a user can move 85
$\xi$ SOM input feature vector 25
$\rho_{\text{pos}}$ position density 57
$\sigma$ standard deviation 35
$\tau$ elapsed continuous time 85
$\tau_i$ ToA of a signal at $x$, departing from BS$_i$ 12

Continued on next page
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau_{ij}$</td>
<td>difference between $\tau_i$ and $\tau_j$</td>
<td>12</td>
</tr>
<tr>
<td>$\phi$</td>
<td>adaptation width in space</td>
<td>30</td>
</tr>
<tr>
<td>$\psi$</td>
<td>adaptation width in power for SPLL</td>
<td>64</td>
</tr>
</tbody>
</table>
Appendix B

Proofs

B.1 MMSE for $x$

In Sect. 2.3.4 it is stated that, for the expected value of the mean squared error $E[(x_M - x)^2]$, the value that minimises it is given by $x = E[x_M|p_M]$ (see (2.10) and (2.11)).

This demonstration shows how to find a minimum for $E[(x_M - x)^2]$ using signal study and algebraic manipulations.

First sum and subtract the term $E[x_M|p_M]$:

$$E[(x_M - x)^2] = E[(x_M - E[x_M|p_M]) + (E[x_M|p_M] - x)]^2,$$  \hspace{1cm} (B.1)

and develop the squares:

$$E[(x_M - x)^2] = E\left[(x_M - E[x_M|p_M])^2\right] + E\left[(E[x_M|p_M] - x)^2\right]$$

$$+ 2E\left[((x_M - E[x_M|p_M]) \cdot (E[x_M|p_M] - x))\right].$$  \hspace{1cm} (B.2)

Using the property $E[E[A|B]] = E[A]$, the term $iii$ turns into:

$$iii = E\left[2E \left[((x_M - E[x_M|p_M]) \cdot (E[x_M|p_M] - x))|p_M\right]\right].$$  \hspace{1cm} (B.3)
As \((E[x_M|p_M] - x)\) is constant, it can be taken out of the inner expectation:

\[
iii = E \left( (E[x_M|p_M] - x) \cdot 2E \left[ (x_M - E[x_M|p_M])|p_M \right] \right) = E \left( (E[x_M|p_M] - x) \cdot 2 \left( E[x_M|p_M] + E[ - E[x_M|p_M]|p_M] \right) \right) = E \left( (E[x_M|p_M] - x) \cdot 2 \left( E[x_M|p_M] - E[x_M|p_M] \right) \right)
\]

which makes the term \(iii = 0\). Then, going back to (B.2):

\[
E[(x_M - x)^2] = E \left( (x_M - E[x_M|p_M])^2 \right) + E \left( (E[x_M|p_M] - x)^2 \right) _{iii} + 0 = E \left[ (x_M - x)^2 \right] + 0,
\]

and the value for \(x\) that minimises \(E[(x_M - x)^2]\) is:

\[
x = E[x_M|p_M]
\]

**B.2 Exponential Filter**

In Theorem 3.2 it is stated that, for an exponential filter of the form:

\[
y_{k+1} = y_k \cdot (1 - a) + a \cdot u_k,
\]

where \(y_k\) is the filter output, \(a\) is the filter parameter and \(u_k\) the filter input signal. This demonstration using induction shows that, if \(0 < a < 1\), and \(u_k = u\), constant \(\forall k\), then \(y_k\) reaches \(u_k\) asymptotically and exponentially.

Considering that \(y_0 = 0\), then the first terms of (B.9) are written as:

\[
\begin{align*}
y_1 & = y_0 \cdot (1 - a) + a \cdot u = (a) \cdot u \\
y_2 & = y_1 \cdot (1 - a) + a \cdot u = (2a - a^2) \cdot u \\
y_3 & = y_2 \cdot (1 - a) + a \cdot u = (3a - 3a^2 + a^3) \cdot u,
\end{align*}
\]
which remain unchanged by summing and subtracting 1 from the term multiplying $u$:

$$y_1 = (1 - 1 + a) \cdot u = (1 - (1 - a)) \cdot u \quad \text{(B.13)}$$

$$y_2 = (1 - 1 + 2a - a^2) \cdot u = (1 - (1 - a)^2) \cdot u \quad \text{(B.14)}$$

$$y_3 = (1 - 1 + 3a - 3a^2 + a^3) \cdot u = (1 - (1 - a)^3) \cdot u \quad \text{(B.15)}$$

In this way the term $y_k$ can be written as:

$$y_k = (1 - (1 - a)^k) \cdot u, \quad \text{(B.16)}$$

which can be verified by applying (B.16) into (B.9):

$$y_{k+1} = \left( (1 - (1 - a)^k) \cdot u \right) \cdot (1 - a) + a \cdot u \quad \text{(B.17)}$$

$$= (1 - a - (1 - a)^{k+1} + a) \cdot u \quad \text{(B.18)}$$

$$= (1 - (1 - a)^{k+1}) \cdot u \quad \text{(B.19)}$$

which closes the induction.

The asymptotic behaviour is observed at the limit of (B.16), making $k \to \infty$:

$$\lim_{k \to \infty} y_k = \lim_{k \to \infty} \left( (1 - (1 - a)^k) \cdot u \right) = u \cdot \left( 1 - \lim_{k \to \infty} (1 - a)^k \right) \quad \text{(B.20)}$$

As $0 < a < 1$, then $0 < (1 - a) < 1$, and therefore $\lim_{k \to \infty} (1 - a)^k = 0$, which gives:

$$\lim_{k \to \infty} y_k = u \quad \text{(B.21)}$$

### B.3 Area of $\text{Pr}(x_q)$ in $\mathbb{R}$

In Sect. 4.2.2 it is stated that the function

$$g(x|x_{k-1}) = \begin{cases} 
    h \cdot \exp \left( -\frac{(r_{k-1} - R)^2}{2\sigma^2} \right) & \text{if } r_{k-1} \geq R \\
    h & \text{if } r_{k-1} < R
\end{cases} \quad \text{(B.22)}$$

with

$$r_{k-1} = \sqrt{(x - x_{k-1})^T(x - x_{k-1})} \quad \text{(B.23)}$$
has unit area if

\[ h = h_{1D} = \frac{1}{2R + \sigma \sqrt{2\pi}}, \]  

(B.24)

To prove this it is necessary to evaluate the gaussian integral:

\[ \int_{-\infty}^{\infty} e^{-x^2} \, dx, \]  

(B.25)

which has no primitive function.

However, the definite integral can be evaluated considering

\[ \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-(x^2+y^2)} \, dx \, dy \]  

(B.26)

Using Fubini’s theorem, (B.26) is equivalent to

\[ \left( \int_{-\infty}^{\infty} e^{-x^2} \, dx \right) \cdot \left( \int_{-\infty}^{\infty} e^{-y^2} \, dy \right) = \left( \int_{-\infty}^{\infty} e^{-x^2} \, dx \right)^2 \]  

(B.27)

On the other hand, applying polar coordinate transformation to (B.26), it follows that

\[ \int_{0}^{2\pi} \int_{0}^{\infty} r \cdot e^{-r^2} \, dr \, d\theta = 2\pi \int_{0}^{\infty} r \cdot e^{-r^2} \, dr \]  

(B.28)

Using the substitution \( r^2 = s \) then \( 2r \, dr = ds \) and

\[ 2\pi \int_{0}^{\infty} \frac{1}{2} \cdot e^{-s} \, ds = \pi \int_{0}^{\infty} e^{-s} \, ds = -\pi \cdot e^{-s}|_{0}^{\infty} = -\pi \cdot (e^{-\infty} - e^{0}) = \pi, \]  

(B.29)

which, when applied to (B.27) leads to

\[ \left( \int_{-\infty}^{\infty} e^{-x^2} \, dx \right)^2 = \pi \]  

(B.30)

And finally

\[ \int_{-\infty}^{\infty} e^{-x^2} \, dx = \sqrt{\pi} \]  

(B.31)

Returning to (B.22), let \( h \) be fixed as an unknown constant.

The area \( A \) under (B.22) is calculated as:

\[ A = \int_{-\infty}^{\infty} g(x|x_{k-1}) \, dr \]  

(B.32)
which is divided in three parts, such that

\[ A = A_i + A_{ii} + A_{iii} \]

with

\[ A_i = \int_{-\infty}^{-R} h \cdot \exp \left( -\frac{(r + R)^2}{2 \cdot \sigma^2} \right) \, dr \quad \text{if } r \leq -R \]  

\[ A_{ii} = \int_{-R}^{R} h \, dr \quad \text{if } -R < r \leq R \]  

\[ A_{iii} = \int_{R}^{\infty} h \cdot \exp \left( -\frac{(r - R)^2}{2 \cdot \sigma^2} \right) \, dr \quad \text{if } r > R \]

Making the substitution \( \frac{r + R}{\sigma \sqrt{2}} = s \) on (B.33) and \( \frac{r - R}{\sigma \sqrt{2}} = s \) on (B.35), then \( dr = \sigma \sqrt{2} \, ds \) for both. Then the sum \( A_i + A_{iii} \) can be written as

\[ A_i + A_{iii} = \int_{-\infty}^{0} h \cdot \sigma \sqrt{2} e^{-s^2} \, ds + \int_{0}^{\infty} h \cdot \sigma \sqrt{2} e^{-s^2} \, ds = \int_{-\infty}^{\infty} h \cdot \sigma \sqrt{2} e^{-s^2} \, ds \]

and using the result from (B.31)

\[ A_i + A_{iii} = h \cdot \sigma \sqrt{2} \sqrt{\pi} = h \cdot \sigma \sqrt{2\pi} \]  

The solution of (B.34) is trivial since

\[ A_{ii} = \left. \int_{-R}^{R} h \, dr \right|_{-R}^{R} = h \cdot R - (-h \cdot R) = 2R \cdot h \]  

Then \( A \) can be finally written as

\[ A = A_i + A_{ii} + A_{iii} = h \cdot \sigma \sqrt{2\pi} + 2R \cdot h \]  

Making \( A = 1 \) gives

\[ h = \frac{1}{2R + \sigma \sqrt{2\pi}} \]

### B.4 Volume of Pr(\( \mathbf{x}_q \)) in \( \mathbb{R}^2 \)

In Sect. 4.2.2 it is stated that the function
\[ g(x|x_{k-1}) = \begin{cases} \left( \frac{r_{k-1} - R}{2\sigma^2} \right)^2 \cdot h & \text{if } r_{k-1} \geq R, \\ h & \text{if } r_{k-1} < R \end{cases} \] (B.41)

with

\[ r_{k-1} = \sqrt{(x - x_{k-1})^T(x - x_{k-1})} \] (B.42)

has unit volume if

\[ h = h_{2D} = \frac{1}{2\pi \cdot \sigma^2 + R \cdot \sigma \sqrt{2\pi^3 + \pi \cdot R^2}}, \] (B.43)

The volume \( V \) of a solid of revolution based on a function \( f(x) \), rotated about the \( y \) axis is given by:

\[ V = \int_0^\infty 2\pi \cdot x \cdot f(x) \, dx \] (B.44)

Returning to (B.41), let \( h \) be fixed as an unknown constant, then the volume can be calculated in two parts, such that \( V = V_i + V_{ii} \), with

\[ V_i = \int_R^\infty 2\pi \cdot r \cdot h \cdot \exp \left( -\frac{(r - R)^2}{2 \cdot \sigma^2} \right) \, dr \quad \text{if } r \geq R \] (B.45)

\[ V_{ii} = \int_0^R 2\pi \cdot r \cdot h \, dr \quad \text{if } r < R \] (B.46)

The solution of (B.46) is rather trivial since it represents the volume of a cylinder with height \( h \) and radius \( R \):

\[ V_{ii} = 2\pi \cdot h \cdot \int_0^R r \, dr = \pi \cdot h \cdot R^2 \bigg|_0^R = \pi \cdot R^2 \cdot h \] (B.47)

(B.45) on the other hand requires a little more calculus to solve:

\[ V_i = 2\pi \cdot h \cdot \int_R^\infty r \cdot \exp \left( -\frac{(r - R)^2}{2 \cdot \sigma^2} \right) \, dr \] (B.48)

Making the variable substitution \( r - R = t \), then \( dx = dt \) and

\[ V_i = 2\pi \cdot h \cdot \int_0^\infty (t + R) \cdot \exp \left( -\frac{t^2}{2 \cdot \sigma^2} \right) \, dt \] (B.49)
\[ V_i = 2\pi \cdot h \cdot \int_0^\infty t \cdot \exp\left(-\frac{t^2}{2 \cdot \sigma^2}\right) \, dt + 2\pi \cdot R \cdot h \cdot \int_0^\infty \exp\left(-\frac{t^2}{2 \cdot \sigma^2}\right) \, dt \quad \text{(B.50)} \]

For the first term, a new variable substitution is needed so that \( t^2 = s \). Then \( 2tdt = ds \) and

\[ V_i = \pi \cdot h \cdot \int_0^\infty \exp\left(-\frac{s}{2 \cdot \sigma^2}\right) \, ds + 2\pi \cdot R \cdot h \cdot \int_0^\infty \exp\left(-\frac{t^2}{2 \cdot \sigma^2}\right) \, dt \quad \text{(B.51)} \]

\[ V_i = -2\pi \cdot \sigma^2 \cdot h \cdot \exp\left(-\frac{s}{2 \cdot \sigma^2}\right) \bigg|_0^\infty + \frac{2\pi \cdot R \cdot h \cdot \sigma \sqrt{2\pi}}{2} \quad \text{(B.52)} \]

where the second term is the result of a Gaussian integral, for which there is no primitive. Its value is calculated in the previous section.

\[ V_i = 2\pi \cdot h \cdot \sigma^2 + R \cdot h \cdot \sigma \sqrt{2\pi^3} \quad \text{(B.53)} \]

So the total volume is given by \( V_i \) and \( V_{ii} \):

\[ V = V_i + V_{ii} = 2\pi \cdot h \cdot \sigma^2 + R \cdot h \cdot \sigma \sqrt{2\pi^3} + \pi \cdot R^2 \cdot h \quad \text{(B.54)} \]

Making \( V = 1 \), then \( h \) must be:

\[ h = \frac{1}{2\pi \cdot \sigma^2 + R \cdot \sigma \sqrt{2\pi^3} + \pi \cdot R^2} \quad \text{(B.55)} \]
Appendix C

Acronyms

AoA  Angle of Arrival
BSOM  Bayesian Self Organising Map
CRT  Cathodic Ray Tube
DECT  Digital Enhanced Cordless Telecommunications
DPM  Dominant Path Model
FM  Feature Map
GPS  Global Positioning System
GSM  Global System for Mobile communications
IR  Infra Red
kNN  k Nearest Neighbours
LAN  Local Area Network
LIDAR  Light Detection and Ranging
LM  Linear Model
LOS  Line-of-Sight
ML  Maximum Likelihood
MMSE  Minimum Mean Squared Error
NaN  Not-a-Number
NLOS  Non Line-of-Sight

Continued on next page
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN</td>
<td>Nearest Neighbours</td>
</tr>
<tr>
<td>ODE</td>
<td>Ordinary Differential Equation</td>
</tr>
<tr>
<td>RADAR</td>
<td>Radio Detection and Ranging</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RM</td>
<td>Radial Model</td>
</tr>
<tr>
<td>RMS</td>
<td>Root Mean Square</td>
</tr>
<tr>
<td>RSS</td>
<td>Received Signal Strength</td>
</tr>
<tr>
<td>SLAM</td>
<td>Simultaneous Localisation and Mapping</td>
</tr>
<tr>
<td>SLL</td>
<td>Simultaneous Localisation and Learning</td>
</tr>
<tr>
<td>SOM</td>
<td>Self Organising Map</td>
</tr>
<tr>
<td>SONAR</td>
<td>Sound Navigation and Ranging</td>
</tr>
<tr>
<td>SPLL</td>
<td>Simultaneous Probabilistic Localisation and Learning</td>
</tr>
<tr>
<td>TDoA</td>
<td>Time Difference of Arrival</td>
</tr>
<tr>
<td>ToA</td>
<td>Time of Arrival</td>
</tr>
<tr>
<td>VQ</td>
<td>Vector Quantisation</td>
</tr>
<tr>
<td>WiMax</td>
<td>Worldwide Interoperability for Microwave Access</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
</tbody>
</table>
Bibliography


[29] Marie Cottrell, Jean-Claude Fort, and Gilles Pagés. Two or three things that we know about the Kohonen algorithm. In *Proc. of the 2nd European Symposium on Artificial Neural Networks: ESANN’94*, pages 235–244, Brussels, Belgium, April 1994.


125


IEEE International Conference on Pervasive Computing and Communications: PerComp’03, pages 143–150, Fort Worth - TX, USA, March 2003.


Lebenslauf

Name: Bruno Betoni Parodi
Geburtsdatum 13.02.1978
Geburtsort Rio de Janeiro - Brasilien
Familienstand ledig
Nationlität brasilianisch/italienisch

Ausbildung
Jan 2001 - Dez 2002 Master of Science der Elektrotechnik an der Bundesuniversität von Rio de Janeiro (UFRJ), Brasilien.
Okt 2002 - Okt 2004 Master of Science der Informationstechnik an der Hochschule Mannheim, Deutschland.

Auszeichnungen/Stipendien
Feb 2001 - Feb 2002 Stipendium von CAPES (brasilianisches akademisches Forschungsdienst) im Rahmen des Masterkurses an der UFRJ, Brasilien.

Fortsetzung an nächster Seite
Feb 2002 - Feb 2003 Stipendium von FAPERJ für ausgezeichnete Leistungen im Rahmen des Masterkurses an der UFRJ, Brasilien.


Berufserfahrung


Aug 1999 - Okt 1999 Werkstudent, Siemens AG (ICN), München. Entwicklung einer GUI in der Programmiersprache JAVA.

Jul 2003 - Okt 2003 Werkstudent, Siemens AG (ICN), München. Entwicklung von Testsripts für Validierung eines VoIP-SIP Servers in den Skriptsprachen IPSL und PERL.

Mär 2004 - Aug 2004 Masterarbeit, Siemens AG (CT-IC4), München. Entwicklung eines Verfahrens zur Signalanalyse in MatLab und JAVA.

Feb 2005 - Jan 2008 Doktorarbeit, Siemens AG (CT-IC4), München. Forschung und Entwicklung neuer Verfahren zur Kalibrierung von Indoor Ortungssystemes in MatLab und JAVA.